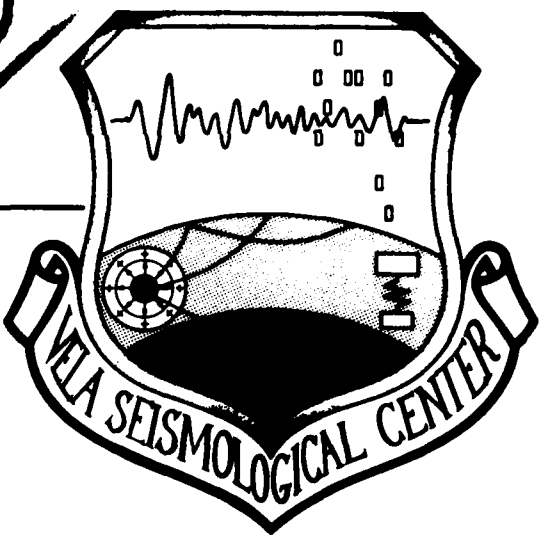
LEVEL II

12

VSC-TR-81-7

EVALUATION OF THE COMMUNICATIONS CONTROL PROCESSOR



W.C. Dean, D.M. Miller and R.P. Colella

Seismic Data Analysis Center
Teledyne Geotech
314 Montgomery Street
Alexandria Virginia 22314

7 MAY 1981

Monitored By:

VELA Seismological Center
312 Montgomery Street
Alexandria, VA 22314

Sponsored by

The Defense Advanced Research Projects Agency (DARPA)

DARPA Order No. 2551

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: VSC-TR-81-7	2. GOVT ACCESSION NO.: AD-A099 286
3. RECIPIENT'S CATALOG NUMBER:	
4. TITLE (and Subtitle): EVALUATION OF THE COMMUNICATIONS CONTROL PROCESSOR	5. TYPE OF REPORT & PERIOD COVERED: Technical Rept.
6. PERFORMING ORG. REPORT NUMBER: SDAC-TR-79-6	
7. AUTHOR(s): W. C. Dean, D. M. Miller, R. P. Colella	8. CONTRACT OR GRANT NUMBER(s): F08606-78-C-0007, ARPA Order-3552
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Teledyne Geotech, 314 Montgomery Street, Alexandria, Virginia 22314	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: VT/8709
11. CONTROLLING OFFICE NAME AND ADDRESS: Defense Advanced Research Projects Agency, Nuclear Monitoring Research Office, 1400 Wilson Blvd., Arlington, Virginia 22209	12. REPORT DATE: 23 October 1979
	13. NUMBER OF PAGES: 98
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): VELA Seismological Center, 312 Montgomery Street, Alexandria, Virginia 22314	15. SECURITY CLASS. (of this report):
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE:

16. DISTRIBUTION STATEMENT (of this Report)

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

Author's Report Date 10/22/79

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Computer Evaluation
Pluribus
Multibus
Multiprocessor
Computer Reliability
Communications Processor
Seismic Data Analysis Center
SDAC

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report evaluates the role of the Pluribus, a multiprocessor/multibus system, in its role as the communications control processor (CCP) at the Seismic Data Analysis Center (SDAC) over the 26 months from its installation in April 1977 through May 1979.

In capacity the six processor Pluribus at the SDAC provides a bandwidth somewhat greater than 750,000 instructions per second, equivalent to three processors working full time. For the 16.8 KB/sec input and 28.2 KB/sec output data streams at SDAC operating in May 1979 this capacity is in excess of

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

double the bandwidth required.

The hardware reliability of the Pluribus lives up to its billing. However, the system suffered over 8% downtime, primarily due to software problems. The CCP is in an R&D environment at SDAC which required revisions of the CCP inputs and functions over the 26 months of operation studied.

Other functions in addition to its CCP functions, such as data recording and signal processing, are not recommended for the Pluribus. Although the Pluribus has the wide bandwidth and the high reliability required for these functions, it was designed primarily for communications and so lacks a file management system necessary to support disks and tapes for data recordings. It also lacks a higher level language. As such the Pluribus is a difficult computer to program. Consequently, generating multiple versions of signal detection algorithms (to enable testing one against another) would be expensive and apt to have an adverse impact on the system reliability.

EVALUATION OF THE COMMUNICATIONS CONTROL PROCESSOR

SEISMIC DATA ANALYSIS CENTER REPORT NO.: SDAC-TR-79-6

AFTAC Project Authorization No.:	VELA T/8709/PMP
Project Title:	Seismic Data Analysis Center
ARPA Order No.:	2551
Name of Contractor:	TELEDYNE GEOTECH
Contract No.:	F08606-78-C-0007
Date of Contract:	01 October 1977
Amount of Contract:	$2,875,210
Contract Expiration Date:	30 September 1978
Project Manager	Robert R. Blandford (703) 836-3882

P. O. Box 334, Alexandria, Virginia 22313

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

ABSTRACT

This report evaluates the role of the Pluribus, a multiprocessor/ multibus system, in its role as the communications control processor (CCP) at the Seismic Data Analysis Center (SDAC) over the 26 months from its installation in April 1977 through May 1979.

In capacity the six processor Pluribus at the SDAC provides a bandwidth somewhat greater than 750,000 instructions per second, equivalent to three processors working full time. For the 16.8 KB/sec input and 28.2 KB/sec output data streams at SDAC operating in May 1979 this capacity is in excess of double the bandwidth required.

The hardware reliability of the Pluribus lives up to its billing. However, the system suffered over 8% downtime, primarily due to software problems. The CCP is in an R&D environment at SDAC which required revisions of the CCP inputs and functions over the 26 months of operation studied.

Other functions in addition to its CCP functions, such as data recording and signal processing, are not recommended for the Pluribus. Although the Pluribus has the wide bandwidth and the high reliability required for these functions, it was designed primarily for communications and so lacks a file management system necessary to support disks and tapes for data recording. It also lacks a higher level language. As such the Pluribus is a difficult computer to program. Consequently, generating multiple versions of signal detection algorithms (to enable testing one against another) would be expensive and apt to have an adverse impact on the system reliability.

TABLE OF CONTENTS

List of Acronyms & Abbreviations

ALK	Alaskan seismic sites
ARPA	Advanced Research Projects Agency, now DARPA
BBN	Bolt, Beranek, & Newman, Inc.
BLSI	Buffered Serial Line Interface
CANSAM	The Canadian Seismic Network
CCA	Computer Corporation of America, contractor for the ARPANET mass store.
CCP	Communication Control Processor
DPS	Detection Processing System at SDAC, also referred to as DP.
HF	High frequency seismic data around 60 Hz resonance
HLC	Compatible Local Host Interface
Hz	Frequency units in Hertz or cycles per second
IBM	International Business Machines, Inc.
IMP	Interface Message Processor
INMOD	Input module, application software in CCP
LASA	Large Aperture Seismic Array in Montana
LP	Long period, refers to data instruments peaked near 20 seconds period (.05 Hz).
MP	Middle period seismic data around 4 Hz resonance
NEIS	National Earthquake Information Service
NEP	Network Event Processor at SDAC
NORSAR	Norwegian Seismic Array near Oslo, Norway
NSS	National Seismic Stations, an experimental unattended seismic site.
OUTMOD	Output module, applications software in CCP
PWY	Pinedale, Wyoming seismic observatory
SAQ	Signal Arrival Queue of the Detection Processing System at SDAC.
SDAC	Seismic Data Analysis Center operated by Teledyne Geotech in Alexandria, Virginia.
SIP	Seismic Interface Processor, at the ARPANET mass store in Cambridge, Massachusettes.
SP	Short period, refers to data instruments peaked near 1 Hz
VELANET	The VELA Uniform Seismic Network
YKA	Yellowknife Array in Northwest Territory, Canada

1 CCP EVALUATION

1.1 Problem Definition

The objective of this study is to evaluate the Pluribus in its role as a CCP at the SDAC and to explore its possible future roles in an expanded seismic network.

The Pluribus is a multiprocessor used as the Communication Control Processor (CCP) in the on-line (VELA) seismic network at the Seismic Data Analysis Center (SDAC) in Alexandria, Virginia. The Pluribus was designed as a modular machine intended for use in packet-switch networks such as the ARPANET. In its simplest version, the Pluribus would be smaller and cheaper than previously used Honeywell 316 and 516 Interface Message Processor (IMP's), while being expandable to provide ten times the bandwidth and five times as many input-output (I/O) devices as the 516 IMP.

At the SDAC the Pluribus, as the CCP, receives incoming seismic data from the phone line modems and the ARPANET, performs various data quality checks and some reformatting, drives a quality control display, combines the several phone line inputs into one data block, and sends this data block to the recording computer, an IBM 360/40. In addition, it transmits some of the incoming data over the ARPANET to the ARPANET mass store at the Computer Corporation of America (CCA) in Cambridge, Massachusetts, as well as transmitting data to Patrick AFB. This report is an evaluation of how well the CCP has performed this role at the SDAC.

Another phase of this evaluation concerns the future role of the Pluribus. Presently, plans are being considered to expand the seismic network in terms of the number of on-line and off-line seismic stations, volume of input data, and reporting schedules for the earthquake bulletin. One possibility is for the Pluribus to continue its present function as a data switch in the expanded network. Another possibility is for some or all of the communication protocol functions to be taken over by microprocessors connected to each modem. Still another possibility is for the Pluribus to take on signal processing and signal detection functions and possibly data recording functions in addition.

This report presents a description of the CCP in Chapter 2, including the architecture of the Seismic Data Analysis System, followed by a description of the Pluribus hardware, its system software, and its application software as a data switch in the seismic network. Chapter 3 estimates the number of

seismic inputs and outputs the current CCP can handle (i.e. its bandwidth) and how many inputs it could handle when fully expanded to 16 processors. In addition, Chapter 3 describes the results of some tests to verify those estimates of CCP bandwidth. Chapter 4 describes the operational and maintenance history of the CCP over the first 26 months of service at SDAC. Chapter 5 compares the Pluribus with other computers of similar and of different architectures for doing the communication functions. Chapter 6 discusses various functions in addition to the communications tasks which the Pluribus CCP could take on and whether such an expanded role for the CCP is recommended.

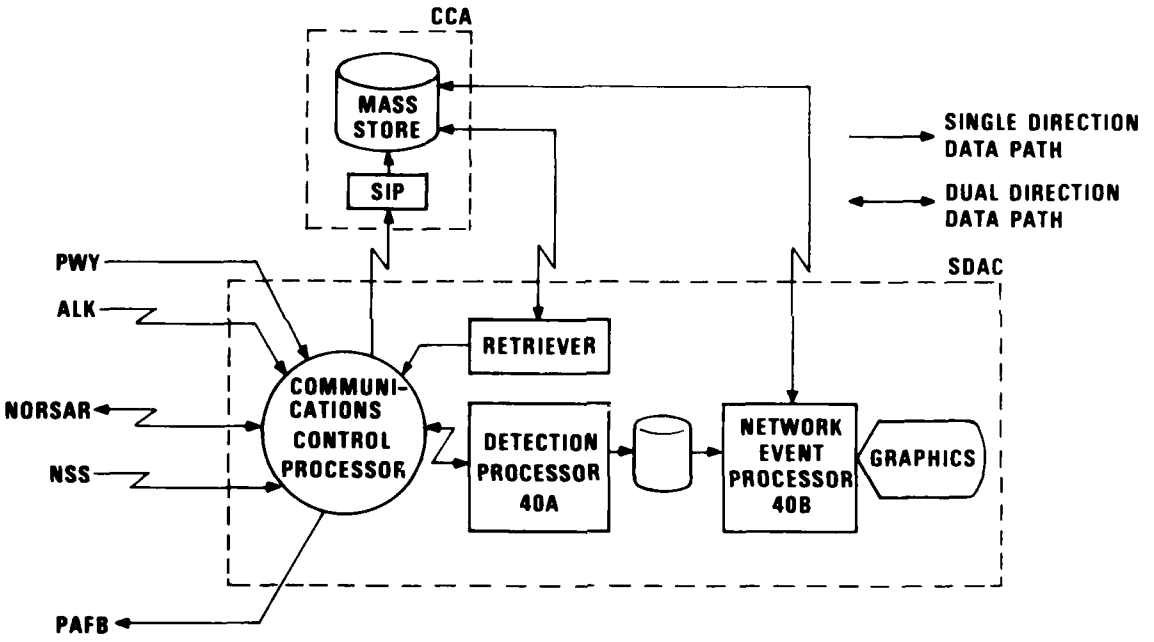


Figure 1.1 Schematic Map of the VELANET

This Page
Intentionally
Left Blank.

2 CCP DESCRIPTION

2.1 SDAC Architecture

The primary role of the CCP is to act as a data switch between the on-line seismic network inputs and the recording and analysis computers at SDAC.

The SDAC receives data from several seismic observatories and transmits raw and processed seismic data as well as seismic event bulletins to the mass store on the ARPANET. Figure 2.1 shows the CCP as the hub of the SDAC on-line system. Acting as a data switch, the CCP receives input seismic data from Pinedale, Wyoming (PWY, 4.8 kilobaud) which includes a channel from Albuquerque, New Mexico; from several sites in Alaska (ALK, 4.8 kilobaud); and from the seismic array in Norway (NAO, 2.4 kilobaud) which arrives via the ARPANET. In addition, a set of seismic channels arrives over an experimental satellite communications link from Tennessee (NSS, 4.8 kilobaud).

The CCP also receives processed lists of detected signals from other seismic observatories. The signal arrival queue (SAQ/EPX) arrives from NORSAR via the ARPANET, intermixed with the real-time data stream, and contains both alphanumeric and signal waveform data. Signal detection lists from the Yellow Knife Array in northwest territories, Canada, from CANSAM (the Canadian Seismic Network), from a United Kingdom set of observatories, and from some of the National Earthquake Information Services (NEIS) stations are keypunched and sent via the IBM 360/44 and the CCP to PAFB. These lists are alphanumeric only and arrive at SDAC in a variety of ways: telephone connections to SDAC terminals, hard copy printouts, and on cards.

Data transmitted to sites include: all real-time short period (SP) and long period (LP) data to IBM 360/44 (DP computer) for recording and detection processing (DP), selected real-time LP channels (Alaskan and NORSAR) to the Seismic Interface Processor (SIP) at the CCA mass store via the ARPANET, and selected real-time SP and LP data and processed data to PAFB.

The key functions of the CCP are:

-- to receive input data from multiple sites in multiple formats using the appropriate protocols for the various sites,
-- to reformat inputs to a common format, and
-- to forward different subsets of the data to several destinations.

Reformatting operations include: converting 12-bit incoming data to 16-bit data for DP use, providing DP and the SIP with a well-defined record format according to the VELANET protocol definition, and converting 16-bit incoming data to 12-bit data for PAFB.

The quality control functions of the CCP involve several tasks. It makes polycode or check-sum verifications and a time-of-day (T.O.D) check on the incoming data streams. It accumulates internal counts (traps) on both the T.O.D. and polycode failures for future statistical analysis. It also keeps message flow counts for all real-time input data streams and keeps count of the number of times system queues are flushed in restart operations. It keeps a world map of the status of individual input channels from each site. A visual control of the quality of the incoming data is available. Up to eight selectable channels of the incoming data may be displayed on a CRT. It monitors the status of the input stations and output sites such as the SIP at the mass store on the ARPANET. Finally it alerts the operator by typing messages on the console, lighting indicator lights, and ringing alarms when polycode, T.O.D., or any of the above conditions indicate trouble, such as a loss of data or a communication path.

Key functions of the on-line system at the SDAC, which the CCP does not perform, include: signal detection or other signal processing, data recording on disk or tape, data retrieval from disk or tapes, or transmission of its own internal state-of-health statistics to other computers. In addition the CCP provides no user support--such as compilers, assemblers, or batch processing for off-line computations.

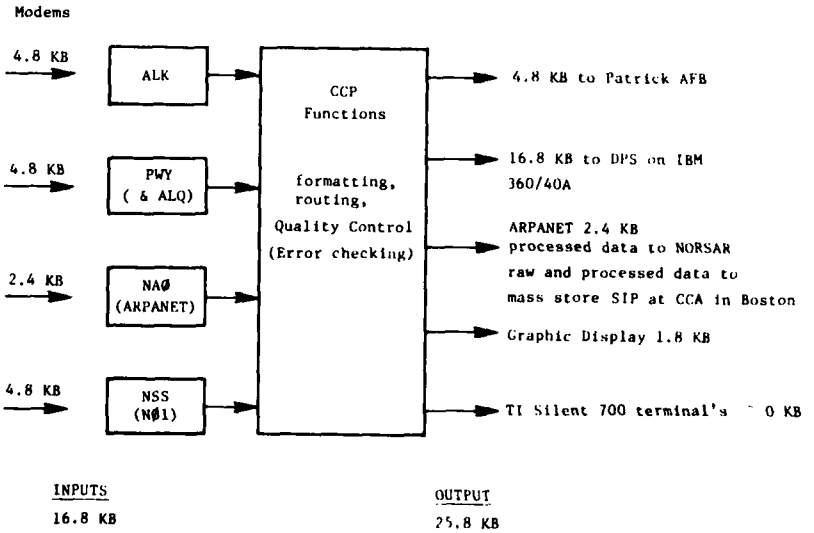


Figure 2.1 Block diagram of current CCP inputs and outputs at SDAC.

2.2 CCP Hardware

The CCP has multiple processors, multiple buses, multiple memories, and multiple I/O ports with the ability to connect each subsystem to every other subsystem.

The Pluribus is a symmetric, tightly coupled multiprocessor, with the modules physically isolated to protect against common failures and with a form of distributed switch employed for inter-module communications. The description of the Pluribus in this and the following sections is abstracted from the paper by Katsuki, et al, (1978). Readers are referred to that paper for more details.

Processor Symmetry

In the Pluribus all processors are identical. In such systems the advantages of redundancy and flexibility are much easier to achieve because they include only one type of processing unit. Even without explicit redundancy, a symmetric system can provide graceful degradation of throughput when a processing element fails. Pluribus systems, which are sized for fully redundant operation, include only one extra processing module; thus, the degradation that results from failure of any processing module consists only of a loss of excess throughput capacity.

Processor Coupling

In the Pluribus the phrase "tightly coupled" means that all processors can access all system resources and perform all parts of the operational program. They operate independently, except for necessary software interlocks on specific I/O devices and data structures.

Flexibility

To permit recovery after failures the Pluribus was designed with at least two of every resource. However, because not all applications required or could afford a fully redundant system, the architecture had to be flexible in at least two ways: 1) in size, the goal was to smooth large incremental steps in the cost-performance curve by utilizing a highly modular design; and 2) in flexibility, the goal was fault-tolerance and fault-recovery, which should be independent of application throughput requirements. Also implied in each of these goals was the requirement for easy expansion.

Multiprocessors based on the unified bus are both easily expandable and conceptually simple structures, but they are vulnerable to single failures anywhere along the bus. In addition, the maximum throughput of such multiprocessors is limited by the design bandwidth of the bus and by contention for common resources. To avoid these problems the Pluribus uses a unified bus to create the functional modules that make up the system, but not to form the main connection structure. The Pluribus has three basic functional modules that share a common address space yet have separate intermodule communications paths: processor buses, memory buses, and I/O buses.

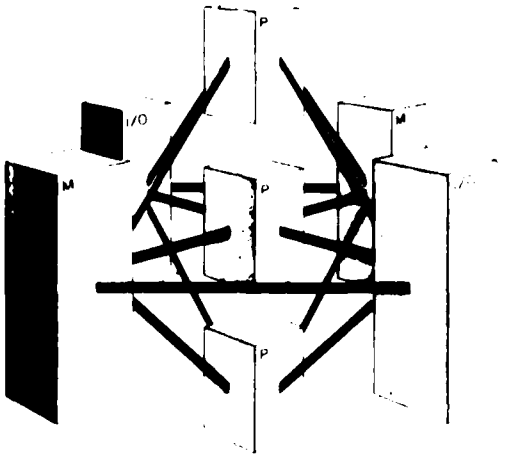


Figure 2.2a A simplified view of the functional modules in a typical Pluribus system showing their interconnectivity. No physical relationships are implied (from Katsuki, et al, 1978).

Modularity

Flexibility and symmetry are achieved by segmenting the operational tasks into strips of code--task distribution routines, task-oriented application routines, and timers--run by any available processor. The code is both re-entrant and accessible to all processors at all times. Primarily, the common memory modules provide space for data buffers, program work areas, and interprocessor communications areas. Code storage is divided into two parts: 1) lightly used code is stored in common memory and is shared between processors; and 2) heavily used code is replicated in local memory for each processor. This strategy minimizes contention for access to common

memory while holding down costs, because, in most applications, only a small part of the code is heavily used. The I/O modules were intended to support both polled low-speed I/O devices and high-speed interfaces capable of direct memory transfers. Couplers provide direct paths both from processor buses to I/O buses for control and polling, and from I/O buses to memory buses for direct memory transfers.

All normal processor-to-processor communication occurs through locations in common memory. However, to initialize the system, it must be possible for one processor to access the local memory and control registers of a processor on a different bus. To allow this, the bus couplers provide a limited reverse path through any common I/O bus.

Physical System Structure

The Lockheed SUE is the minicomputer in the Pluribus. It is a 16-bit machine, similar to the DEC PDP-11; however, in contrast to the PDP-11, the bus arbitration logic is physically separated from the processor. This feature permits a bus to have one or several processors, or none at all. The Pluribus uses the bus, arbitration logic, processors, memories, and several minor I/O units of the SUE.

Bus Structure

The basic Pluribus building block is the bus module. This module contains a modified SUE bus and card cage for up to twenty-four cards, together with completely self-contained cooling fans and power supply. Two bus modules can be connected to form an extended bus.

A processor bus may contain one to four processors and their associated local memories, a bus arbiter, and one bus coupler per logical path. The CCP has 8K words of local memory for each processor. Bolt, Beranek and Newman have determined that the optimum is two processors per bus, which is the arrangement in the CCP. The common memory bus contains an arbiter, bus coupler cards for all the connected paths, and enough memory modules to support the application. The CCP has 128K words of common memory.

In addition to the bus arbiter and bus coupler cards, an I/O bus also contains cards for each of the various types of I/O interfaces that are required, including interfaces for modems, terminals, and host computers, as well as interfaces for standard peripherals. The I/O bus also houses a

number of special units, including: 1) a real-time clock (RTC) used by the system for timing processes and communications links; 2) a special hardware task disbursing unit known as the pseudo-interrupt device (PID) (discussed further below); and 3) a reload card that monitors up to eight communication lines, watching for and processing specially formatted reload messages from the outside world. The CCP does not contain the reload cards.

Inter-Bus Connection System

Because all processors in our system must be able to perform any task, buses are connected so that all processors can access all shared memory and control the operation and sense the status of any I/O unit. To connect processors and common memory, one card of a bus coupler is installed on a common memory bus, the other on a processor bus. Similar connections are made from every processor bus to every common I/O bus.

A block diagram of the CCP is shown on Figure 2.2b and in more detail on Figure 2.2c. The complement of equipment in the CCP includes the following:

TABLE 2.2

Processors

- 6 Lockheed SUE computers, 2 each on processor buses 10, 12, and 14.

Memory

- 6 local memories of 8K words (16 bits), each one devoted to one of the SUE processors on buses 10, 12, and 14
- 2 common memories of 48K words, one each on memory buses 16 and 26.

I/O Units

- 12 Buffered Serial Line Interfaces (BSLIs), 2 each per BSLI card.
- 2 Direct Memory Access/Compatible Local Host's (DMA/HLCa) for high speed data transfers to the Detection Processor (DPS) and to the ARPANET IMP
- 2 Texas Instrument Silent 700 terminals, one on each I/O bus.
- 2 Time-of-Day (T.O.D) clock interfaces, one on each I/O bus.
- 1 Tektronix CRT display for quality control seismic channels, connected to Processor Bus 12.
- 1 Console Control Panel and Indicator lights connected to Processor Bus 12.

Miscellaneous Units

- 2 Pseudo Interrupt Devices (PIDs), one connected to each memory bus.
- 2 Real Time Clocks, one connected to each memory bus.
- 2 check sum devices (DMA/CBT), one connected to each memory bus.
- 14 Bus Couplers for connection between each pair of processors, memory and I/O buses. Each Processor Bus is coupled to each memory Bus. In addition, one memory bus is coupled to the other memory I/O bus.

Pseudo-Interrupt Device

Real-time systems employ priority interrupt mechanisms to direct the attention of the processor to the most urgent tasks. Reliability and load

sharing requirements necessitate that any processor be able to service any I/O device; it also raises questions such as which processor to interrupt for servicing. In the Pluribus, each "interrupt event" (e.g. MDA completion, RTC tick, or software events)--instead of actually interrupting a processor--writes a value associated with its priority to a hardware queuing device called the PID. The software is designed to allow each processor to put aside periodically the content of its present computation and check the PID. When read, the PID will produce the highest value that has been stored in it and, simultaneously, will delete that value from its internal queue. The processor can then use that value as an index to a table of tasks to be performed. The software uses the PID in a similar manner: each time a "strip" of code is completed, it writes the number of the next strip in that task to the PID. When that becomes the highest number in the PID, the next available processor will execute the associated strip.

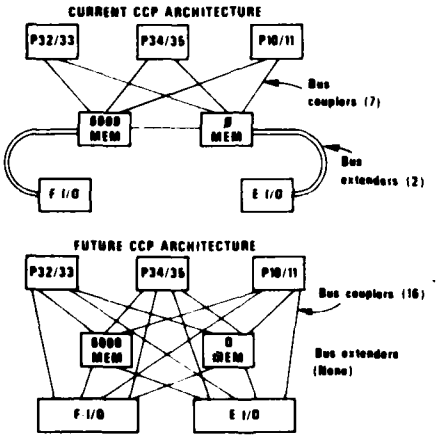


Figure 2.2b Simplified Block Diagram of the CCP Showing the Present and Planned Connections between Processor, Memory and I/O Buses.

In the CCP the 'E' memory and 'E' I/O buses are connected electrically through a bus extender system. The two 'F' buses are similarly connected. As a result when an I/O bus is in use, its memory bus cannot be accessed and vice versa. The memory and I/O buses could be independent of each other, and in a completely redundant Pluribus they would be. To change the CCP to this form would require elimination of the bus extender cards and the addition of 9 bus couplers for a total of 16. All memory units would have to be connected to the memory buses rather than I/O buses. Also, all DMA units would have to be connected to the I/O buses rather than the memory buses.

SDAC CCP CONFIGURATION

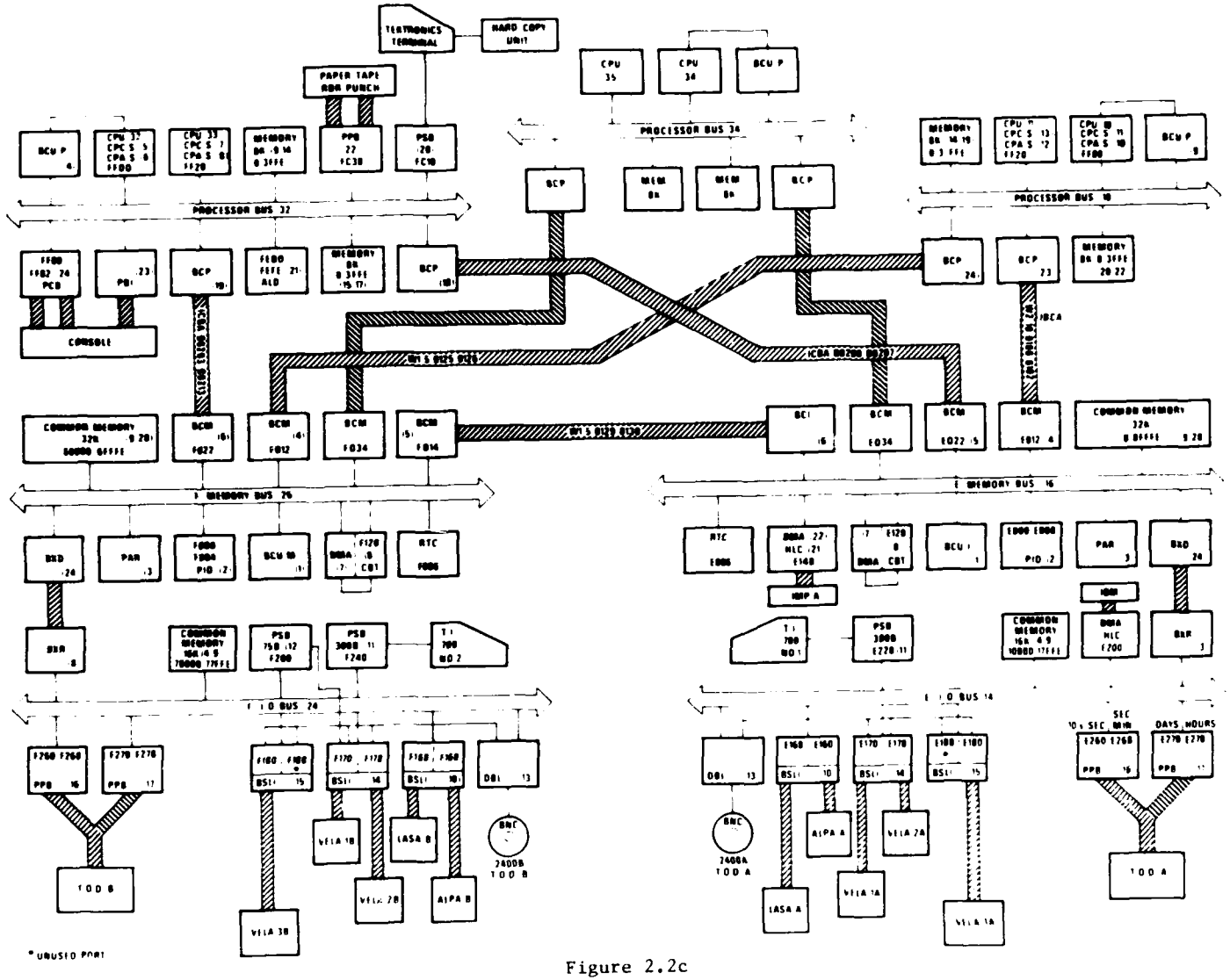


Figure 2.2c

2.3 The Pluribus Operating System

Unlike most conventional systems, principal responsibility for maintaining reliability in the Pluribus lies with the system software rather than in the hardware structure.

The Pluribus hardware was designed to provide an appropriate vehicle for software reliability mechanisms. In addition to normal error checking and reporting in the hardware itself, programmed tests using known data patterns are run at intervals. When hardware errors are detected, system software exploits the redundancy of the hardware by forming a new logical system configuration that excludes the failing resource, using redundant counterparts in its place.

Pluribus systems also check the validity of their software structures. Redundant information is intentionally introduced into the data structures at various points and checked by processes operating upon those structures.

General Responsibility of the Operating System

The software reliability mechanisms for a Pluribus system are coordinated by a small operating system (STAGE) that manages the system configuration and the recovery functions. The operating system is intended to maintain a reliable, current map of the available hardware and software resources.

The operating system must monitor its own behavior, as well as the behavior of the application system. The system cannot assume that any element of hardware or software is working properly--each must be tested before it is used and retested periodically to ensure that it continues to function correctly.

Hierarchical Structure of the STAGE System

The Pluribus operating system is organized as a sequence of stages that are polled by a central dispatcher. A processor starts with only the first stage enabled. As each stage succeeds in establishing a proper map of its segment of the system state, it enables the next stage to run. Each stage uses information guaranteed by earlier stages and thus runs only if the previous stage has successfully completed its checks. Once enabled, a stage will be polled periodically to verify that the conditions for successful completion of that stage continue to apply. The system applies most of its processing power to the last stage that is enabled, but returns periodically

to poll each earlier stage. The application system is the final stage in the sequence and runs only after the earlier stages have verified all the configuration information of the application and the validity of the data structures.

Table 2.3 lists each stage of the Pluribus operating system, together with the aspects of the environment it guarantees. Because processors continue to perform each of the stages periodically, changes in the environment will eventually be noted. Any stage detecting a discrepancy in the configuration map will disable all later stages until the discrepancy is repaired. Then, all the later stages, which might depend on data verified by the disabling stage, will be forced to run all their checks, guaranteeing that they will make any further modifications to the configuration map necessitated by the first change. A serious failure, such as a nonexistent-memory interrupt, disables all but the first stage. In these cases, some reconfiguration might be needed, and all stages should perform all their checks before the application system is resumed.

TABLE 2.3

Stage	Function
0	Checksum local memory code (for stages 0, 1, 2). Initialize local interrupt vectors and enable interrupts. Discover Processor bus I/O. Find some real-time clock for system timing.
1	Discover all usable common memory pages. Establish page for communication between processors.
2	Find and chechsum common memory code (for stages 3, 4, 5). Checksum whole page ("reliability page").
3	Discover all common buses, PIDs, and real-time clocks.
4	Discover all processor bus couplers and processors.
5	Verify checksum (from stage 2) of reliability page code (for rest of stages plus perhaps some application routines). External reloading of missing code pages is possible once this stage is running.
6	Checksum all of local code.
7	Checksum common memory code. Maintain page allocation map.
8	Discover common I/O interfaces.
9	Poll application-dependent reliability and initialization routines. Periodically trigger restarts of halted processors.
10	Application system.

Pluribus operating system stages, from Katsuki et al. (1978).

Establishing Communication

Since all processors in the Pluribus must perform the same checks, they must agree about the state of the system resources. Coordinating multiple processors with potentially different views of the hardware configuration requires two mechanisms: 1) the processors must agree on an area of common memory in which to record the machine configuration map, and 2) they must cooperate in their decisions to modify the map.

The first step in coordinating the multiple processors of a Pluribus is to agree on a page of memory for communications. Prior to establishing the page, the processors have no way to communicate about where it will be located. The procedure must operate correctly in the face of failures that might leave some of the processors seeing a different set of common-memory pages from the rest. Processors that are unable to see the communication area will attempt to use another memory page and must be prevented from interfering with unaffected processors.

Any processor that is first starting up (or restarting after some massive failure) can assume nothing about the location of the communication page. Any page can be used; therefore, a small area for communication control variables is reserved on each page of common memory. In addition, each processor must maintain its own view of the system in its own local memory. Once all (or most) processors have agreed upon a communications page, they will attempt to establish a global view of the hardware environment. Every processor attempts to establish the lowest number (lowest address in memory space) page that it sees as the page through for communication. To be valid, any page must have a pointer to the current communication page, and the communication page's pointer must point to itself.

If the memory bus containing the communication page is lost, all processors will attempt to establish a new communication page on the other bus. Using their timers on the new lowest page, which initially points to the old one after the failure, they await the threshold. No one is holding the timers to zero, so the new page becomes the communication page when some processor's timer first runs out.

The Consensus Mechanism

When configuration data must be updated, the Pluribus processors accomplish this goal by a consensus. Each stage has a consensus maintained as part of its environment. The first step in forming a consensus is to determine the set of processors that is executing the corresponding stage. This set has certified the primitives necessary to successfully maintain this stage's portion of the configuration map. For the system to respond to failures, the consensus must be kept current--new processors must be able to join it rapidly, and processors that may have halted or ceased to run the stage must be erased from the set.

Because each processor in the Pluribus performs each stage of the checking code, the consensus mechanism provides the coordination needed to change the configuration map gracefully. When a stage detects a failure, the processor sets the appropriate fix-it bit and disables the following stages. When enough processors detect the failure, they implement the fix to the configuration map. Now these processors can complete the later stages, devoting their attention to any further changes required by the failure. A processor seeing a different picture of the resources and unable to reach agreement with the rest of the system hangs indefinitely at the point of detecting the discrepancy. The technique effectively prevents the processor from damaging the system.

Application-Dependent Checking

Ideally, the application system will perform its own checks before initiating or resuming normal operation. The last stage provides a mechanism that polls application-oriented processes to perform consensus-driven checks and repairs of their own data structures. This stage uses the results of the hardware (application-independent) discovery stages to certify its own data structures. For example, it could allocate or de-allocate device parameter blocks as the I/O devices are discovered or disappear and initialize spare memory pages for use as data buffers as they become available. User-written reliability checks can be performed on the application data structures, and the appropriate reinitialization invoked to remedy failures.

2.4 Application Software

The application software in the CCP handles its multiple inputs and processing responsibilities by a process-queue structure in which each queue provides temporary storage for the subsequent process.

The continual data input streams are divided into one-second blocks. Since it is a real-time system, the CCP must handle all quality control, reformatting, and transmission functions on this total (one-second) block of data within one second, including sufficient time for all overhead functions, in order to be ready for the next one-second block of data. All of the CCP functions are divided into a sequential string of small processes separated by queues. The queues reside in memory and act as temporary data storage for the next process. The queues are written with one-second blocks, with an average queue three seconds long.

The input from each seismic observatory passes through four processes in the CCP as shown in figure 2.4. The first process is the input line interface. The functions performed by this module are line protocols, line synchronization, and byte reversals. A typical input site arrives over a 4.8 kilobaud line. The output of this process enters the raw data queue.

The raw data queue is drained by the Input Module for that particular observatory. This module performs checksums, time of day (T.O.D) verification, and status updates. This module also performs 12-to-16 bit conversions as necessary, since all seismic data in the SDAC system is in 16-bit words, whereas some data is generated at the observatories in 12-bit words. The output from this input module fills the processed queue for this site.

The Process Data Queue has data for all input sites awaiting processing by output modules. The average queue length is two to three seconds of data per site. The input data message will remain in the Process Data Queue until all output modules requiring this message copy the data out. At that time, the memory holding this message is returned to the free packet list.

An Output Module, using a parameterized selection list, collects data from the Processed Data Queue to be delivered to a particular output site. In the current SDAC application there are several output modules, using different output protocols to deliver data to the DPS, to the SIP over ARPANET, to the local graphics display, and to PAFB. There are two high

level protocols. The VELANET protocol (Briscoe, 1977) is used for data to the SIP and the DPS which allows selection of data type (SP, LP, MP, and HF) and station. A de facto protocol (Texas Instruments, 1975) is used for the PAFB link which allows selection of individual channels from any station. The output modules, using their specific protocols, will build output messages from selected input data streams and place them on the appropriate output queue.

An output module will attempt to gather data in time sequence for its particular output site. If the data from one seismic input station for a given second has not yet arrived at the Processed Data Queue, the module will wait a predetermined amount of time for that data before proceeding. If at the end of the predetermined wait that data still has not arrived, the output module will finish gathering the other input data for that second and then proceed with gathering data for the next second.

The output queues are drained by the line driver for that specific output interface. The line drivers take care of the lowest level protocols such as a bisync-type protocol to PAFB or the ARPANET host/IMP protocol to the SIP.

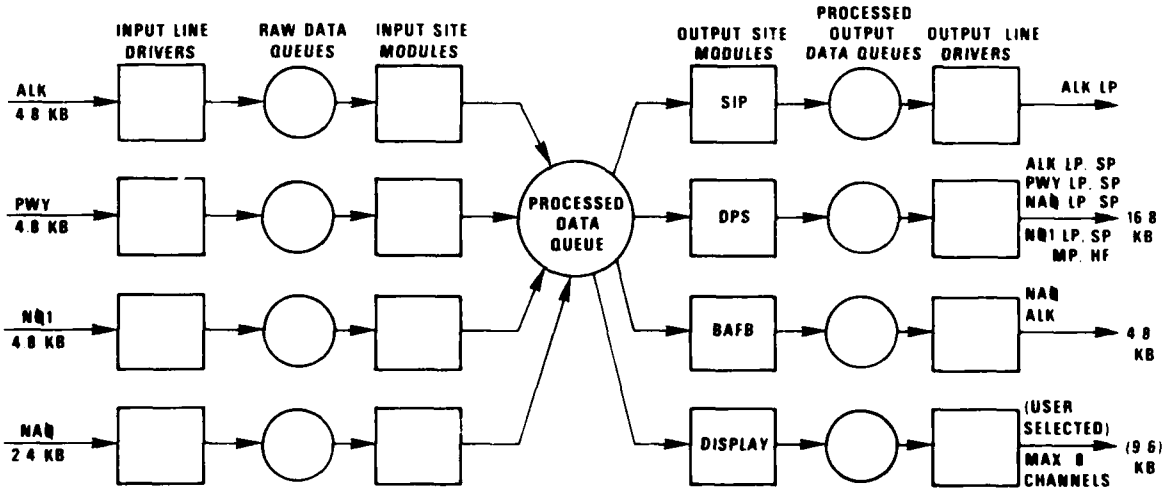


Figure 2.4 Architecture of the Applications Software in the CCP.

This Page
Intentionally
Left Blank.

3 CCP CAPACITY

3.1 Calculated CCP Bandwidth

The bandwidth required for the CCP functions, when measured as a percentage of the capacity of a single processor, includes approximately 60% for overhead tasks, 9.6% for each 4.8 kilobaud input and 6.1% for each 4.8 kilobaud output data stream.

By bandwidth on a computer, we mean the number of instruction cycles per second available for data processing tasks. In the Lockheed SUE computers the average instruction takes 4 microseconds so that there are 250,000 instruction cycles available per processor in the CCP. As in all Pluribus installations, the bandwidth of the CCP processor, memory, and I/O ports in the CCP must be far higher than that required by the input and output data streams so that the CCP continues to function normally when several components are not working.

In this section we will give estimates of the bandwidth required by the tasks to be done in the applications and background software. In the following section we will describe the results of some tests to measure the actual system bandwidth by disabling processors and increasing the number of input or output data streams to be processed. When the remaining processors become overloaded, the STAGE system will shut off data flow and restart operations, thereby providing a quantitative measure of its actual bandwidth.

We calculated the bandwidth requirements on the CCP by estimating the number of instruction cycles required for each background and application task and multiplying by the repetition rate expected for each one.

Table 3.1a lists the background tasks required in the CCP with the repetition rate and the execution time of each. The processes are typical background software tasks in a Pluribus. STAGE is the operating system explained in section 2.3. The task, JIFFY, is an interrupt handler with a 60-Hertz repetition rate. DISPLAY is a handler which answers the interrupt from the Tektronix display unit delivered when it has finished processing its previous character. Its maximum repetition rate is 1200-Hertz. Both JIFFY and DISPLAY are normal interrupt routines in that they are answered directly by the processors.

The interrupt behavior of all the remaining background tasks differ from normal interrupts in that they write a value associated with the task priority to the Pseudo-Interrupt Device (PID). Each processor will set aside

its present computation periodically and check the contents of the PID for the next highest priority task. RTC-F is the Real Time Clock-Fast and RTC-S is the Real Time Clock-Slow. These two software modules are used to cycle other software functions at repetition rates which are various multiples of their own (clock) repetition rates. CONFIG is a module which periodically checks the operability of all I/O devices and looks for alternate devices if some are found not working. RESTART is the lowest priority task and functions as the logical extension of the PID. Typically it is used to delay reactivating low priority tasks. EMTY provides an indication that a PID is empty and redirects the system's attention to the next PID.

IDLE is used simply as a quality control measure indicating how much the CCP is idle. We make use of it in testing the level of saturation of CCP activity when some of the processors are purposely taken off-line.

All processors must perform JIFFY and STAGE. Hence, for all other functions, the 250,000 cycles per second of each processor is reduced to 247,612 cycles per second (or 99%). Exclusive of JIFFY and STAGE, the total demand in background tasks, for which all processors will share computational responsibility is 72,425 instruction cycles per second, or 28.97% of the capacity of one processor. Including JIFFY and STAGE, one processor would require 30% of its capacity to handle all overhead tasks by itself.

The last background task is an application idling function. If there is no seismic data flowing through the CCP, then the processors will wake up the processes, check whether data is present, do some minor bookkeeping duties, and put the processes back to sleep. With no data present, the bandwidth required for this application idling will be 97,200 instruction cycles per second. With seismic data flowing through the CCP, there will be less idling activity since the processor will be busy with data processing. With a data flow equivalent to the current SDAC system, we estimate that the application idling activity will be only 75 percent of its maximum or approximately 73,000 instruction cycles per second (29% of one processor's capacity).

The application software requires a bandwidth proportional to the data flowing through the CCP. As described in section 2.4, the input data from a typical seismic site will pass through four processes in the CCP: an Input Line Driver, an Input Site Module, an Output Site Module, and an Output Line

Driver. The input modules handle data from only one site. The output modules can handle some or all of the data from every site. Figure 2.4 shows the data flow.

For a 4.8 kilobaud seismic input, similar to that from the Alaskan sites, the Input Line Drivers (BSLIs) will require 8550 and the Input Site Modules 15500 instruction cycles per second. These estimates are obtained by calculating the time required--by counting the number of instructions--for each software routine and multiplying by the repetition rate for the routine to complete processing on a one-second block of seismic data. The overhead prior to setting up the application software loops is trivial so that these estimates will vary linearly with the bandwidth of the input data channel.

Demand estimates for the output Modules and Output Line Drivers are obtained in a similar way. For a data stream of 4.8 kilobaud the Output Module will require 15000 and the Output Line Driver 250 instruction cycles per second. Table 3.1a summarizes these results.

TABLE 3.1a
System 'Background' Tasks
Actual Task Run Time and Cycle Usage

Task Name	Label		Repetition Rate	Exec Time	Instruction Cycles	Cycles/Second
STAGE	T_S		9/sec	528 μsec	132	1,188
JIFFY	T_J		60/sec	80 μsec	20	1,200
						2388
DISPLAY	T_D	(max)	1200/sec	45 μsec	11.25	13,500
RTC-F	T_{TF}		600/sec	328 μsec	82	49,200
RTC-S	T_{TS}		40/sec	332 μsec	83	3,320
CONFIG	T_C		1.25/sec	528 μsec	132	165
IDLE	T_I	(max)	40/sec	256 μsec	64	2,560
RESTART	T_R		40/sec	196 μsec	49	1,960
EMTY	T_E		40/sec	172 μsec	43	1,720
						72,425

Process	Instruction Cycles/Second	Percent Capacity of one Processor
Unshared Background Tasks		
JIFFY and STAGE (by every processor)	2388	1.0%
Shared Background Tasks	72.425	29.0%
Applications Idling Function	73,000	29.0%
Applications:		
Input modules per 4.8KB data stream	24,050	9.6%
Output modules per 4.8KB data stream	15,250	6.1%

Background tasks will not vary; 1% for every processor plus 50% shared equally by all operating processors.

Applications tasks are proportional to the input or output data streams.

We can use these estimates to calculate the load required for the current network. However, there are three I/O ports which do not match the 4.8 kilobaud standard. One is the input via the ARPANET from NORSAR (NAØ) which has a 2.4 kilobaud bandwidth but enters via a DMA rather than the BSLI, as do the other inputs. Its load requirements are 7.4% of the processor. The second is the 4.8 kilobaud line to PAFB which leaves via a BSLI rather than a DMA, as do the other outputs. Its load requirements are estimated to be 10% of one processor. Finally, there is the 1.8 kilobaud special interface to the QC display which acts like a BSLI output and requires 7% of one processor. Using these special I/O estimates together with the standard ones we can approximate the total application and background load for the current network as follows (Table 3.1b):

TABLE 3.1b

INPUTS	% Capacity of 1 processor
ALK at 4.8 KB via BSLI	9.6%
PWY at 4.8 KB via BSLI	9.6%
NØ1 at 4.8 KB via BSLI	9.6%
NAØ at 2.4 KB via DMA from the ARPANET	7.4%
OUTPUTS	
to NAØ at 2.4KB via DMA over the ARPANET	3.0%
to DPS at 16.8 KB via DMA (6.1% * 16.8/4.8)	21.4%
to SIP at 2.4 KB via DMA over the ARPANET	3.0 %
to PAFB at 4.8 KB via BSLI	10 %
to Display CRT at 1.8 KB via special interface	7 %
Total Applications	80.6%
Shared overhead	58 %
Local overhead (1% per processor)	6 %
Total overhead	64 %
Total CCP	144.6%

3.2 Testing for CCP Bandwidth

Measurements show that the CCP will shut down, when subjected to more input and fewer processors, at approximately 50% of the capacity of the working processors due to resource contention conflicts.

We can use the CCP itself to measure its bandwidth. Since the CCP gives positive indications via lights on the front panel whenever it shuts off the data stream and restarts the STAGE system, we have only to increase the input bandwidth, reduce the number of processors operating, or both, to arrive at a condition where the input saturates the processors. We can incrementally increase the input bandwidth by duplicating input data streams. We can incrementally decrease the number of processors operating by powering down one (for two processors) or two (for four processors) processor buses. Hence, we can incrementally increase the load on the operating processors and thereby gain a fair measure of the bandwidth limits of the CCP.

In the bandwidth tests on the CCP, we used from one to four 4.8 kilobaud inputs. The first was that from the Alaskan sites, ALK; the second from Wyoming, PWY; the third was the Alaskan input duplicated, and the fourth was the National Seismic System, NSS input, NØ1. All 4.8KB inputs represent equivalent loads to the CCP.

For outputs during the bandwidth tests we used only two. All inputs were sent to the Detection Processor (DPS) or to both the DPS and the Seismic Interface Processor (SIP) at the mass store. The SIP and DPS discarded the excess data. No other special outputs to Patrick AFB or NORSAR were sent during the tests.

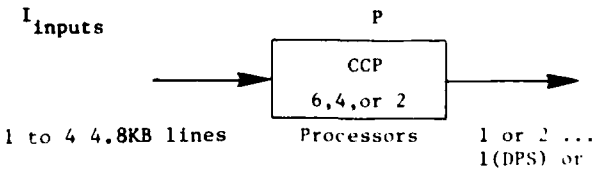
The number of processors is easy to vary by pairs. To remove one pair of processors requires nothing more than removing power to one processor bus by a switch. To remove only one processor on a bus is somewhat more complicated, requiring modification of a software control parameter.

Table 3.2a shows the results of these bandwidth tests. The left column indicates the number of inputs, outputs, and processors in the test. The input bandwidth will be a multiple of 4.8 kilobaud lines as explained above. The output bandwidth equals the input bandwidth for one output and is double the input for two outputs. The IDLE count is a measure of the idle time in the processors. If the IDLE count is high, the processors have a lot of free

TABLE 3.2a

I-O-P	Input BW KB/sec	Output BW KB/sec	IDLE Count	Applications demand %	Background demand %	Total demand %	Load per processor %
1.1.6	4.8	4.8	3539	15.7	58.0	73.7	12.3
1.2.6	4.8	9.6	3786	21.8	58.0	79.8	13.3
2.1.6	9.6	9.6	3331	31.4	58.0	89.4	14.9
2.2.6	9.6	19.2	3358	43.6	58.0	101.6	16.9
3.1.6	14.4	14.4	3973	47.1	58.0	105.1	17.5
3.2.6	14.4	28.8	3530	65.4	58.0	123.4	20.6
4.2.6	19.2	38.4	2767	87.2	58.0	145.2	24.2
1.1.4	4.8	4.8	2095	15.7	58.0	73.7	18.4
1.2.4	4.8	9.6	1982	21.8	58.0	79.8	20.0
2.1.4	9.6	9.6	1545	31.4	58.0	89.4	22.4
2.2.4	9.6	19.2	1608	43.6	58.0	101.6	25.4
3.1.4	14.4	14.4	1936	47.1	58.0	105.1	26.3
3.2.4	14.4	28.8	1350	65.4	58.0	123.4	30.9
4.2.4	19.2	38.4	1210	87.2	58.0	145.2	36.3
1.1.2	4.8	4.8	244	15.7	58.0	73.7	36.7
1.2.2	4.8	9.6	188	21.8	58.0	79.8	39.9
2.1.2	9.6	9.6	153	31.4	58.0	89.4	44.7
2.2.2	9.6	19.2	140	43.6	58.0	101.6	50.8
3.1.2	14.4	14.4	-	47.1	58.0	105.1	52.5*
3.2.2	14.4	28.8	-	65.4	58.0	123.4	61.7*
4.2.2	19.2	38.4	-	87.2	58.0	145.2	72.6*

* Failed



Bandwidth Tests on the CCP

time each second. As the processors become more loaded the idle count decreases. When the processors are saturated with more work than they can complete in a second, the STAGE system shuts the CCP down, flushes the data memories, and restarts. In this case the IDLE count is never taken, and there is no idle time for any of the processors.

The applications demand is the calculated bandwidth requirements for the specified inputs, according to the analysis outlined in section 3.1. The unit of demand is 100% of a single processor's capacity of 250,000 instruction cycles per second. The background demand is measured the same way and is a constant 58% for all experiments. The last column indicates the proportion of applications plus the background demand each operating processor must handle.

The experiments are listed in groups of operating processors. The experiments with six processors are listed first, those with four processors are second, and those with only two processors are last. Within each group the experiments are listed in order of increasing demand.

The results show that the system never became saturated with at least four processors operating. However, the IDLE counts for four processor experiments were significantly less than those for the six-processor experiments.

The system became saturated only on the last three experiments, with two processors operating, and only after the load per processor exceeded 50.8 percent. Note that with a load of 50.8 percent the system continued to function, but with an IDLE count of only 140 versus 3358 when six processors were operating on the same input/output load. Thus, the experiments pinpoint the saturation limit of the CCP within two percent of a single processor's capacity.

The system shuts down when only two processors are operating, with each processor loaded only slightly above 50 percent. However, both of these processors are on the same bus. Even though most of their code is in their own local memory, both use their common bus when they access their own local memory. In addition, both use the same bus to access common memory. Hence, when one of the two processors is working, the other is probably idle waiting for a chance to use the bus.

The system throughput could be significantly increased if the two operating processors were on different buses. These last three experiments, plus a couple of others, were run again with two processors and two processor buses in operation. The results are shown on Table 3.2b. For two inputs and one output, neither the single-bus nor the double-bus arrangement failed, but the IDLE count increased from 153 to 653 with the two processor buses. For three or four inputs and one or two outputs, the single bus arrangement always failed (52.5% or higher loading per processor) whereas the double-bus arrangement never failed (up to 72.6% loading per processor). Moreover, the IDLE counts on these two processor, double processor bus arrangements were higher with multiple inputs and outputs than the single input cases with a single processor bus.

These single/double processor bus comparisons demonstrate that resource contention can limit the CCP throughput. A good rule of thumb will be that the CCP bandwidth will be a single processor's capacity (250,000 instruction cycles per second) times the number of processor buses. For the six-processor CCP, this bandwidth would be approximately 750,000 instruction cycles per second.

BBN argues (BBN staff, 1975) that two processors per bus have a throughput considerably greater than 100% of one processor's capacity. Each 4 microsecond instruction requires only half that time for an instruction fetch, the remainder for execution. As a result the two processors will share the bus by overlapping instruction cycles. In the CCP, however, the bus extenders which link an I/O and a memory bus are slow relative to the processor's instruction cycle and are a main reason for the resource contention problems. Consequently, the rule of thumb quoted above applies to the CCP as it now stands but is expected to improve considerably when the extra bus couplers are added and the memory and I/O buses become independent of each other.

TABLE 3.2b

I-O-P	Input BW KB/sec	Output BW KB/sec	IDLE Count	Applications demand %	Background demand %	Total demand %	Load per processor %
2.1.2	9.6	9.6	653	31.4	58.0	89.4	44.7
3.1.2	14.4	14.4	418	47.1	58.0	105.1	52.5
3.2.2	14.4	28.8	134	65.4	58.0	123.4	61.7
4.1.2	19.2	19.2	476	62.8	58.0	120.8	60.4
4.2.2	19.2	38.4	276	87.2	58.0	145.2	72.6

These examples are for two processors operating on different buses. No failures occurred.

This Page
Intentionally
Left Blank.

4 CCP OPERATIONAL STATISTICS

4.1 Operational History of the CCP

The CCP has suffered over 8% downtime at SDAC, due largely to the R&D environment, resulting in frequent revisions of its functions.

This evaluation covers the 26-month period from April 1977 when the CCP became operational through May 1979. The statistics for the CCP over that period, summarized in Table 4.1, show that downtime averaged 8.39% System crashes are CCP shutdowns due to hardware or software failures. Problem studies are shutdowns by the maintenance or development staff to study long-standing problems which may be causing only momentary loss of data. One such software error caused a loss of 7 seconds of data before each hour. Monthly downtimes are also listed in Table 4.1.

Operations will shut down the CCP to make a copy of the current version of the application software and operating system on paper tape. Normally, the current version of the software is kept on file at two computers in California which are accessible via the ARPANET. If these computers are unavailable for any reason when the CCP must be reloaded, it is necessary to use the backup on paper tape.

System testing is a category including the development of new operating versions, such as when a new seismic site comes on-line, or other revision of the system or applications software. Much of this downtime occurred during the shake-down period in the early months of the CCP operation.

The non-operational software version may or may not be transmitting data currently. Usually the CCP is being operated under a new version of the operating system which is being evaluated prior to its approval.

If the system testing and non-operational software version categories are removed as being part of the CCP development time, and not part of its operational time, then the downtime was only 3.08%. If the operating environment were stable, several of these categories would be significantly reduced. With the operating system unchanging, the operations staff would have no need to shut the system down to copy a new version onto paper tape. System development and testing of new versions would virtually disappear. Presumably, most of the bugs would be out of the system so that problem studies and system crashes would be significantly decreased. Under such conditions the major downtime would be for preventive maintenance, and all downtime might total no more than 1.5%.

TABLE 4.1

CCP Downtimes (in hours:minutes per month)

Month	Preventive Maint.	System Crashes	Problem Studies	Operations (data flushed)	System Testing	Non-operational Version	Monthly Total
1977 Apr	2:54	0:54	-	2:06	39:06	-	45:00
May	-	2.36	-	3:54	18:00	-	24:30
Jun	8:00	8:48	-	4.36	226:42	-	248:06
Jul	11:30	14.24	-	3:24	33:48	-	63:06
Aug	11:18	39:24	-	15:12	117:12	-	183:06
Sep	98:06	61:54	-	9:36	117:18	-	286:54
Oct	18:54	12:24	-	2:06	88:06	-	121:30
Nov	10:00	6:00	-	2:18	39:18	-	57:36
Dec	9:54	4:30	-	8:18	41:12	-	63:54
1978 Jan	1:30	4:24	-	4:12	44:06	-	54:12
Feb	2:06	5:18	-	6:06	7:24	-	20:54
Mar	30:20	26:34	-	-	4:05	-	60:59
Apr	2:05	3:23	-	1:26	13:43	-	20:37
May	-	3:28	-	-	4:18	-	7:46
Jun	8:57	4:50	0:23	-	36:39	-	50:49
Jul	2:28	3:55	0:09	-	9:51	23:47	40:10
Aug	5:17	6:48	0:43	-	3:13	19:34	35:35
Sep	3:03	4:19	0:08	-	1:44	-	9:14
Oct	8:49	6:15	0:04	1:52	11:01	-	28:01
Nov	16:50	8:17	0:38	2:11	11:21	-	39:17
Dec	0:58	4:11	-	-	7:34	18:02	30:45
1979 Jan	-	3:51	0:11	2:22	11:52	-	18:16
Feb	1:20	0:48	0:10	0:38	4:09	-	7:05
Mar	3:13	2:22	0:05	-	37.49	-	43:29
Apr	2:28	1:03	0:04	0:53	9:52	-	14:20
May	2:28	6:46	0:44	-	7:07	-	17:05
Totals	262:28	247:26	3:19	71:10	946:30	61:23	1592:16
% downtime	1.38%	1.30%	0.02%	0.37%	4.99%	0.32%	8.39%

SUMMARY

Cause	Hours	Percent of Total Time
Preventive Maintenance	262.47	1.38
System Crashes	247.43	1.30
Problem Studies	3.32	0.02
Operations (data CCP flushed from system)	71.17	0.37
System Testing	946.50	4.99
Non-operational Software Versions	61.38	0.32
Total CCP Downtime	1592.27	8.39%
CCP downtime exclusive of system and non-operational versions	584.39	3.08%

Summary of CCP downtime from April 1977 through May 1979.

4.2 Software Development and Maintenance History of the CCP

The on-going software development of the CCP, in response to changing requirements, caused considerable downtime and tended to obscure a creditable hardware maintenance record.

The CCP inputs and functions have been changed several times since it was first installed at SDAC. Initially the inputs included LASA and ALPA (long-period) data. Plans were also made for KSRS data which never came. Since then, however, ALPA was shut down, and the Alaskan sites (ALK at 4.8 Kilobaud with both LP and SP data) took over as CCP input. LASA data was discontinued and replaced with PWY. Later an additional channel was added from Albuquerque (ALQ) via the PWY phone link. The connection between the CCP and DPS was originally via the ARPANET. Later this was changed to a direct interface at the SDAC avoiding the ARPANET. Changes were made also in the PAFB line and the NORSAR link over the ARPANET. Finally, the NSS data was added to the input stream. Not only did this data stream represent another site, but it also brought data channels with two new sampling rates which caused major revisions in the software.

All of these changes involved software revisions. Such revisions require testing and debugging. Most of the downtime can be attributed to this development and debugging environment rather than to hardware difficulties with the CCP.

There have been some hardware failures throughout the past 26 months. During that time only six integrated circuit chips were replaced. Troubles from those chips included unexpected PID interrupts, bus hangups, bus power failures, bad sync pulses--all of them intermittent. Another problem occurring occasionally was dirty contacts. On processor cards the dirty contacts show up as a processor which has difficulty loading, or which is shut down occassionally and is attempted to be reloaded by the other processors several times. There is a flaw in the gold alloy in these contacts, and a task of preventive maintenance (PM) now is to periodically clean these contacts with a pencil eraser. Note, however, that the CCP is designed to continue to operate in spite of such hardware problems and in fact continued to do so.

We have noted two problems in design. The first concerns the extended memory/input-output buses which both show an uncalled-for spike on the signal-acknowledge line. The spikes, due to a feedback circuit employed on the bus extender (BXR) cards to delay the signal-acknowledge pulse, occur at a time when the system ignores them. They should not appear at all. The second concerns the bus coupler cards (BCP) which do not employ a deskew map register for the real-address lines. Occasionally these registers are read early and the wrong address is in the register. When this happens the given processor will shut down.

In spite of being a multiple processor system the CCP must still be taken off line for PM. The reason is that bus structure in the CCP is not fully redundant so that all necessary resources for on-line operation cannot be switched to their counterparts on other buses. This situation will change in the future with the installation of the fully redundant bus couplers. In addition the use of a CCP mock-up will permit running all maintenance diagnostics and other tests on CCP cards off-line.

Other hardware problems can appear because the CCP is not completely redundant. We have described the bandwidth limitation due to two processors sharing a common bus (see Section 3.2). Other non-duplicated assets in the CCP include:

1. Two time-of-day (T.O.D) clocks and 2 interfaces, but only one clock is attached at any given time. Hence, if that T.O.D. is on the 'E' bus and 'E' bus goes down, the CCP loses T.O.D.
2. DMA/HLC's to DPS and the IMP are connected to the 'E' bus. If the 'E' bus goes down, these DMA outputs are lost.
3. The paper tape reader/punch and the auto load card are attached to the Processor bus 32 and not duplicated elsewhere.
4. The memory and I/O buses are electrically connected via bus extender cards so that these are only two rather than 4 such buses. Hence, when I/O bus 14 is in use, memory bus 16 is also tied up.
5. The CCP console and front panel (lights, et al) are connected to Processor Bus 32 and not duplicated elsewhere.
6. The Tektronix display is connected to the Processor Bus 32 and not duplicated elsewhere.

4.3 Ease of CCP Programming and Maintenance

The Pluribus, not being a commonly known, mass-produced machine and not having complete documentation, is more difficult to program and to maintain than other computer systems.

The Pluribus is not a mass-produced machine. Due to the intricate hardware and software structure, and the lack of comprehensive documentation, the CCP is difficult to program and maintain.

The multi-processor, multi-bus design of the Pluribus hardware demands sophisticated maintenance personnel. Hardware problems frequently manifest themselves in an indirect manner, requiring insight and experience to locate and fix. BB&N offers an excellent maintenance course on the Pluribus, and we recommend that any new team taking over the CCP maintenance attend the BB&N course. For senior maintenance personnel who have been through this course, maintaining the CCP should not represent an unusually difficult assignment.

Programming the CCP is not an easy task. First, the entire software system, which is written in Pluribus assembler, is always treated as a single program. As a result, the generally understood distinction between an operating system and a program under its control does not hold for the CCP. Its operating system, STAGE, is closely intertwined with the applications code, so much so that even a small mistake in the applications code can bring the entire system down. Furthermore, when such bugs are introduced into the system they are often difficult to find because of the fault-tolerant design of the CCP software. Therefore, it is necessary for the programming staff maintaining the CCP to have a good understanding of how the entire system operates.

All editing of the source file and assembling is done at a remote site via the ARPANET, thus requiring some working knowledge of the network and the remote host computer. This arrangement necessitates the transfer of very large listings over the ARPANET from the remote host to a local host for printing.

These hurdles, taken separately, are not overly significant. Taken together they represent a formidable source of frustration and even confusion. Thus, senior systems programmers are the programming talent

needed for software development and maintenance on the CCP. As in the case of hardware maintenance, we also recommend the software staff be given a one-week course on the Pluribus by BB&N if the CCP is to be installed at a new organization or new programming personnel were to acquire the CCP responsibility.

The multi-processor, multi-tasking nature of the CCP (Pluribus) imposes the restriction that all logical tasks be concurrently re-entrant. This restriction is actually an advantage. For like tasks the actual logic module need only be coded once and only one physical copy need be resident in the system with no requirement to ever refresh the module.

This advantage becomes evident in the case of adding a new data source to the CCP. As long as transmission medium and data format of the new source are the same as a currently defined source, only parameter blocks and task PID values need be defined for the line interface and station input modules. In the case where either the transmission medium or data format of the new source station is different, new logic modules must also be implemented. In addition to the line interface and station input modules, a new station also requires that the following be defined 1) the station's channel names and channel attributes, 2) site and data-type masks for use by the output modules, and 3) queue pointers for both raw and processed data queues.

The total memory requirements for an ALK type station, including message buffering through the system, comes to a total of approximately 3,500 bytes.

Table 4.3 lists each of the hardware changes and each of the software steps required to add one ALK-type of station to the CCP.

TABLE 4.3

Requirements To Add One ALK Type Station

1. Hardware

1.1 External to the CCP

- Lease Line (4.8 KB)
- Modem (2)

1.2 Interval to the CCP

- Serial Interface Card (BSLI) primary
- Serial Interface Card (BSLI) back-up

2. Software

2.1 System (I/O, Type, Config, etc.)

- Device Table Entries (10 tables)	24 Bytes **
- Device Test Routine	
- Device Initialization Routine	
- Device Driver Module Initialization Routine	

2.2 System (Input Module, etc.)

- Input Module Initialization Routine	
- Dispatch Table Entries For:	
SLI Driver; BASPAR, BASE, BASMAP	6 Bytes *
Input Module; BASPAR, BASE, BASMAP	6 Bytes *
- Module Initialization Tables For:	
SLI Driver; INIDIS, S2PBLK	4 Bytes **
Input Module; INIDIS, S2PBLK	4 Bytes **
- Queue Pointer Entries For:	
Raw Data Queue; QSTART, QEND, QLOCK	6 Bytes **
Processed Queue; QSTART, QEND, QLOCK	6 Bytes **
- Actual Queue Space Used:	
Raw Data Queue ~ 1 second of data	640 Bytes ***
Processed data Queue ~ 2.5 seconds of data	1,600 Bytes ***

2.3 Application (Line Driver)

- BSLI Driver Module	
- Parameter Block For Driver Module	38 Bytes **

2.4 Application (Station Input Module)

- Input Module (Common Code)	
- Input Module (Specific Code)	
- Input Module (Status Code)	
- Parameter Block For Input Module	202 Bytes **

2.5 Application (Station Channel ID's, etc)

- CHFORM 1 word/channel	52 Bytes **
- CHSTAT 1 word/channel	52 Bytes **
- CHRAT0 1 byte/channel with status	-
- CHRAT2 1 byte/channel with status	-
- CHRAT3 1 byte/channel with status	-
- CHID1 1 word/channel	52 Bytes **
- CHID2 1 word/channel	52 Bytes **

2.6 Application (Output Modules)

- SELLST (Data Selection List)	2 Bytes **
- Output Message 'Queue' space (~ 1 Msg/sec)	748 Bytes ***

3. Total Memory Requirements

3.1 Static ~ 500 Bytes

3.2 Dynamic ~ 3000 Bytes

3.3 Total 3500 Bytes

* 'Pre-allocated' & static, each module uses 1 'slots'
** Allocated on variables page, static
*** Allocated on buffer pages, dynamic

5 ALTERNATIVE ARCHITECTURES FOR THE CCP FUNCTIONS

5.1 The Three Types of CCP Architectures

There are three basic hardware architectures available for the CCP: a single minicomputer, a pair of minicomputers, and a multi-processor/bus computer.

In this section our primary interest is in alternative hardware for the communications functions of the CCP. In some of the alternatives the hardware responsible for the communications functions can easily handle such functions as recording or signal processing. In other architectures these extra functions are difficult to accommodate. Still in this age of distributed processors, whether a particular architecture can mix communications and other functions in the same computers is not a strong advantage, especially when the seismic network requires multiple processors to handle the load.

There are only three basic types of hardware architectures currently available to handle the communications functions. The first, and simplest, is a single minicomputer handling all inputs. The second, a logical extension of the first, is a pair of minicomputers running in parallel. The third is the multi-processor, multibus architecture represented by the Pluribus.

These three types of systems will be explained in more detail in the following sections. They will also be compared to the Pluribus CCP in reliability, bandwidth, expandability, ease of programming, ease of maintenance, and costs.

5.2 Single Computer Versus the Pluribus

The single minicomputer as CCP is a single point-of-failure system and may be more difficult to expand, but it is cheaper and less complicated than the Pluribus and can be backed up with a duplicate.

Since the seismic data exists in 16-bit formats, the 16-bit minicomputer seems like the simplest choice for the CCP functions. Its role would be to receive all seismic inputs from the modems and ARPANET, perform the polycode and time-of-day checks, keep failure statistics on both of these checks, reformat data as required by DPS, PAFB or other outputs, drive the quality control display, and provide the operator with information and control over all of these functions.

The single mini-computer architecture represents a single point-of-failure system. However, there are ways of improving the probability of its uptime. Since spares are needed for PM and development, these can be placed on line together. If the primary system went down, data acquisition would be switched to the backup system.

The most significant difference between the pluribus and single minicomputers is that the Pluribus is a multibus, fault-tolerant combination of minicomputers. Consequently, it should have a wider bandwidth and be more reliable than a single minicomputer. In hardware reliability the Pluribus should be more reliable than a single minicomputer. Katsuki, et al (1978) report that early tests of the Pluribus showed they exceeded 99.7 percent availability and, more recently, above 99.9 percent. BB&N expects these failure rates to improve as the machines mature. By availability BB&N means operational uptime divided by scheduled uptime excluding power and air conditioning failures.

Unfortunately, power and air conditioning failures plus preventive maintenance and all the software development shutdowns, while not the fault of the Pluribus hardware, nonetheless present a loss of seismic data. These downtimes far exceed the hardware failures of both the Pluribus and a single minicomputer CCP. Thus in such a system, whether the communications control functions are performed by the Pluribus, which is unavailable 0.1 percent of the time, or by a single minicomputer, which is unavailable 1.0 percent of the time, is inconsequential.

With regard to expandability and flexibility it is evident that the general structure of the Pluribus is extremely modular. First, the busses are modular in that a variable number of units can be plugged into each bus. Bus extender units permit busses to be lengthened to accommodate more units. Furthermore, with a modular bus interconnection scheme via separate bus couplers as opposed to a centralized "cross bar" switch, the number of busses in a system can expand or contract to suit needs. Theoretically one could incorporate dozens of processor and memory busses and up to four I/O busses into a system. In most cases the expanding of the hardware structure will require little, if any, change to the operating system. For example, to add two new processors would require the hardware installation of the processor bus, the bus couplers, the processors and their local memories. The software (for our CCP) would require changes only in the few areas which are processor dependent and for a 'pure' Pluribus software system no changes would be required.

Due to this expandability and flexibility in both the hardware and software structures, the Pluribus is more adaptable to a wide range of seismic inputs. For a network of seismic sites, each of which generates an input volume of 4.8 kilobaud, and for an output bandwidth from SDAC of double the input, the following table shows the number of SUE processors needed in the Pluribus to adequately handle the traffic.

No. of Processors in the Pluribus	No. of seismic sites at 4.8 KB/sec each
6	10
8	15
10	19
12	24
14	28
16	33

These estimates were made by the analysis given in Section 3.2. Moreover, all of these expansions could be made with little or no change to the applications software; only parameter modifications would be necessary.

For a single minicomputer system expansion would come in larger increments. The initial minicomputer might have the bandwidth to

accommodate 10 seismic sites. If only 4 sites were on-line, the minicomputer would be under utilized, but that is precisely the situation we see at the SDAC with the Pluribus CCP. If the network expanded beyond 10 sites, another minicomputer system performing the CCP functions for the additional sites would be put in parallel with, but independent of, the original system. Again the application software in the second system could probably duplicate that in the first, except for parameters. Following the two independent minicomputers there would have to be some data concentrator connected with the recording functions.

For ease of programming, ease of maintenance, and ease of learning by new operator and maintenance personnel, the simpler minicomputer system would be advantageous. Certainly minicomputers could be chosen which are in common use, are widely known, and for which comprehensive documentation exists. The Pluribus system is at a disadvantage on all of these points. Still, one can expect the documentation of the Pluribus to improve, and as machines of the multibus architecture become more common, unfamiliarity will be less of a problem.

A comparison of costs for the two systems would favor the minicomputer system, if the seismic network did not expand beyond the capacity of a single minicomputer to handle the traffic. If parallel systems of independent minicomputers were needed, the costs probably would still favor the single minicomputer system, but only slightly.

In software development the costs for the Pluribus system would be higher than for the single minicomputer system. Since all of the functions of the computers checking each other and replacing each other in the Pluribus are carried out by software, the programming in the Pluribus CCP is more complex. If the two or more independent minicomputers were needed to handle the seismic traffic, the software costs of the two types of systems would be more nearly equal but still be somewhat higher for the Pluribus system.

5.3 Dual Minicomputer-CCP's On-line

The single-point-of-failure criticism for a single minicomputer-CCP can be largely overcome by putting a spare minicomputer on-line in parallel with the primary system.

As was stated in the previous section, a single minicomputer architecture represents a single point of failure system. A common solution to this problem is to run two identical systems in parallel. Figure 5.3 shows a block diagram of the arrangement. If the CCP does not perform any recording functions, then the recording computer and some amount of temporary storage (disk drive) would have to be duplicated too. Then, any failure in the primary system permits all data to be recovered, even that which would otherwise be lost before an operator could answer the alarm and get a spare unit operating. Since the data on the back-up system is never saved when the primary system is functioning normally, the temporary storage space for the back-up system need not be very large. The secondary system would take over the primary role during preventive maintenance on the primary system. In practice the identical computers would probably trade primary and secondary roles on a regular basis.

The advantages to this architecture are the same as for a single minicomputer: cheaper and less complex hardware and software, greater availability of trained support personnel and with the additional advantage of increased system reliability through redundancy. In practice the reliability of such a dual minicomputer CCP should approach that of the Pluribus CCP.

The main disadvantage to the dual computer approach is again much the same as for the single computer approach; when bandwidth is exceeded a larger computer must replace the current computer, only now two computers must be upgraded. In addition to the bandwidth problem there is the requirement that both computers remain identical through any hardware or software upgrades that may be required through the life cycle of the system.

The cost of the dual minicomputer CCP will be approximately double that of the single minicomputer-CCP. The dual minicomputer system does not duplicate all hardware (the temporary disk storage is smaller on the secondary system than that on the primary), and it does use much of the same software. However, extra software costs are involved in linking the

two systems together, causing the switch-over to automatic or semi-automatic (i.e. with operator help). Even at double the single minicomputer-CCP, the cost of the dual system should still be somewhat less than that of the Pluribus system.

All versions of the CCP, including the single minicomputer-CCP, have expected up-times which are superior to the availability of the commercial power supply as our experience at SDAC has demonstrated. If the superior on-line (hardware) performance of the Pluribus or the dual minicomputer systems are to be realized, then the commercial power must be backed up by an emergency power supply. Furthermore, if the CCP takes on no recording functions, then the recording computer and its temporary storage (disk drives) must be connected to the emergency power supply also.

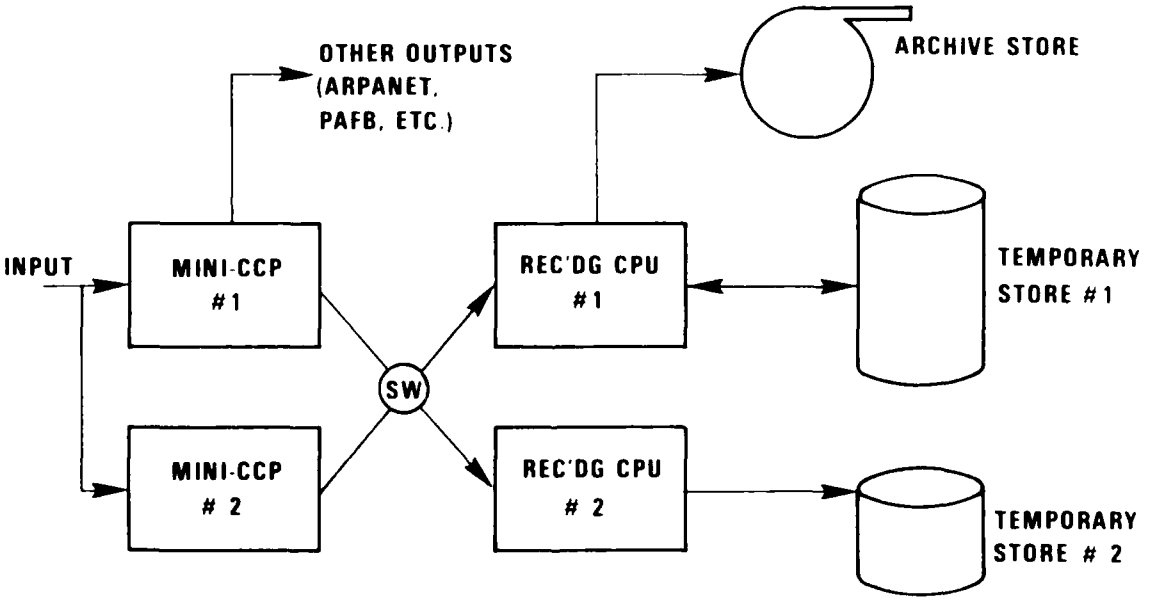


Figure 5.3 A block diagram of a dual minicomputer system for CCP and recording functions.

5.4 Multiprocessor/multibus Architectures

Pluribus competitors, available from Tandem Computers, Inc. and PRIME, are similar in architecture, offer the same fault-tolerant operation, and have a wider bandwidth by virtue of using faster minicomputers than the Pluribus.

The Pluribus architecture is quite distinct from the single mini- or parallel minicomputer architectures for CCP functions. As discussed in the preceding chapters the Pluribus with its multiprocessor/multibus architecture was chosen for the CCP originally because it provides a solution to the problems of bandwidth, reliability, and system expandability.

A commercially available competitor to the Pluribus, built with a similar multibus architecture, which is listed in the October 1978 edition of the DataPro Reports on Minicomputers, [6] is the T16 manufactured by Tandem Computers, Inc.

Tandem Computers began operations in 1974 with the objective of developing and marketing a line of minicomputer systems for applications which depend heavily on the continuous availability of their computers. Furthermore, these systems were to be modular so they could be readily expanded without reprogramming. Tandem adhered to these objectives and in 1976 brought out their current line of Non-stop Systems. These fault-tolerant systems represent Tandem's sole business activity. The Tandem Non-stop Systems use multiple processors, multiple controllers, multiple data paths between the system modules, and multiple power supplies in all system configurations. This design provides both added reliability and added capacity. The higher reliability comes from the multiplexed configurations which offer a high probability that at least one processor and one data path will always be operational. The higher capacity comes from the system design employing the parallel paths and duplicate resources to achieve a higher throughput.

Defective modules can be replaced without powering down the balance of the system. Likewise additional computing power can be introduced via more processors, more memory, or more peripherals without reprogramming or other detrimental effects.

The storage units can include disks and tapes. The disk or tape controllers are connected to dual channels and thus can be powered from either of two processors. Thus extending the CCP functions to include recording will be somewhat easier with the Tandem system than with the Pluribus.

The Tandem computers are faster than the Lockheed SUE computers. They possess an add time of 0.5 μ seconds as opposed to the SUE's 3.0 μ seconds. This bandwidth gain is a result of newer technology (1976 minicomputers versus a 1972 model for the SUE). Presumably a new Pluribus could also be supplied with faster minicomputers and provide equal bandwidth. The Tandem systems are expandable from 2 to 16 processors.

There are over 250 Tandem systems in operation as of October 1978. DataPro reports that users give the system good marks generally, and particularly in maintenance service, technical support, ease of programming, and manufacturer's software. Since these systems represent Tandem's only business, these services might be expected to receive more attention than they would from a firm with many different kinds of business.

Table 5.4 lists a set of hardware which matches that of the CCP at the SDAC. The costs given are those listed by DataPro and do not include any applications software or system integration into the seismic network. Also extra peripherals, such as the CRT display for seismic data quality control, are omitted. Except for the fact that the Tandem system uses faster computers than the CCP (Pluribus) and thus is expected to have four to six times the bandwidth, the performance of the two systems is expected to be comparable. Likewise, the costs of the system hardware are comparable.

TABLE 5.4

6 - T16/1403 CPU's with 96K bytes (48K words) of 0.5μ sec memory @ $22K each CCP memory 6* 8K words = 48K words local + 2* 32K words = 64K words common = 122K words memory	$132K
2 - 74 K byte (32K word) parity core memory	$ 16K
2 - T16/6603 Hard copy terminals, 30 cps, 132 columns, RS-232	$ 6.4K
2 - T16/6202 Byte service controller (4 synchronous communications lines)	$ 11.6K
2 - T16/7501 Terminal Patch Panels	$ 1.6K
2 - T16/7104 Cabinets (for 4 CPU's each)	$ 13.6K
1 - T16/9202 FORTRAN	$ 6.0K
Total	$187.2K

A list of the major hardware components for a Tandem CCP which matches the Pluribus CCP at SDAC.

Source: DataPro Reports on Minicomputers, October 1978.[6]

Another system using a multiprocessor, multibus architecture is the PRIME system. A block diagram is given in Figure 5.4. This system is older than the Tandem but is not listed in DataPro Reports of October 1978. With computers and memories becoming cheaper and more capable, we can anticipate more multiprocessor architectures in the future.

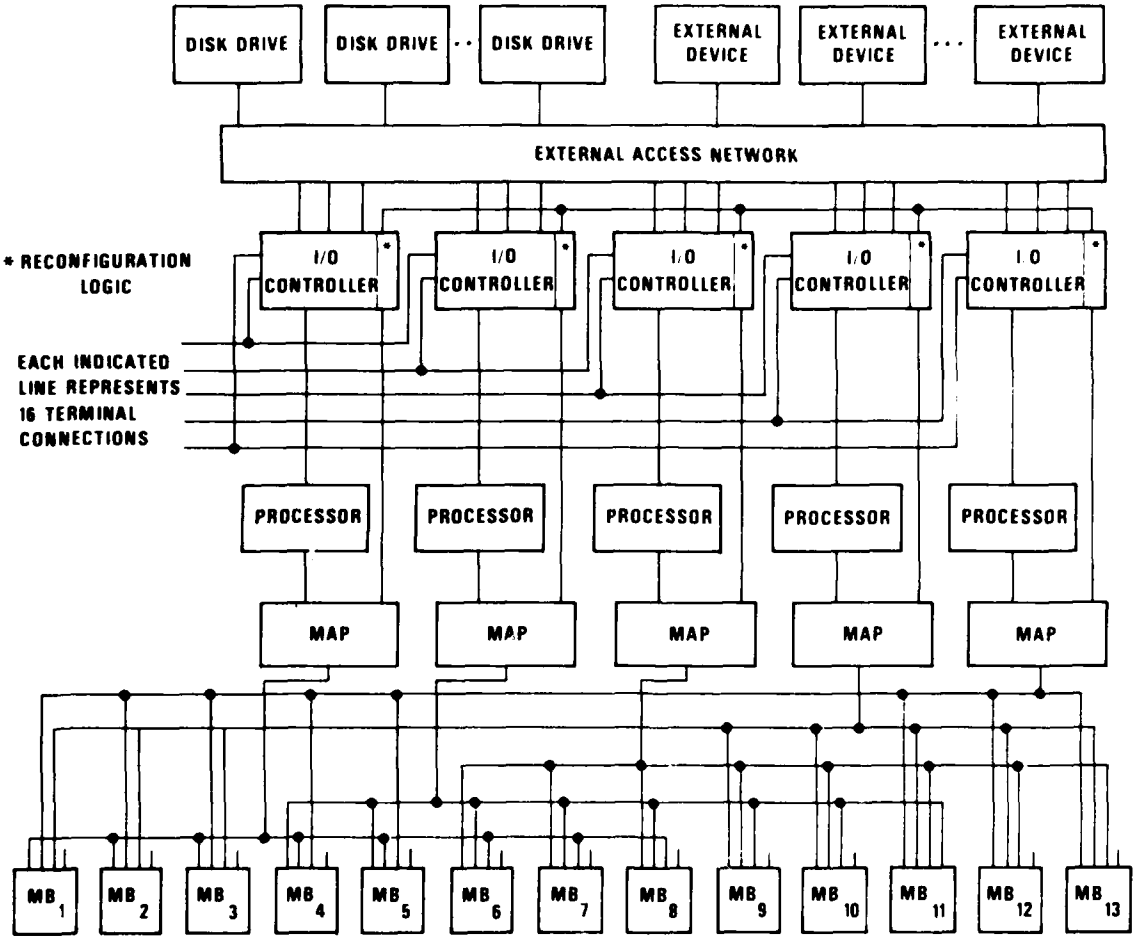


Figure 4 A block diagram of the PRIME system.

6 ALTERNATIVE FUNCTIONS FOR THE PLURIBUS

6.1 Data Recording in the CCP

Although the Pluribus CCP has the wide bandwidth and the high reliability required for data recording, it lacks a file management system necessary to support disks and tapes, a shortcoming which would make the software development costs for recording excessive.

To provide recording in the (SDAC) seismic network, a computer must have a high bandwidth, both for recording and for retrieving the data. In recording the data must flow first into temporary storage such as disks, and periodically from temporary storage into archive storage such as tapes at the SDAC or the mass store via the ARPANET. If the analysis system (NEP) also must access the data while it is being held in temporary storage, the retrieval bandwidth increases further. If the input stream from the seismic network were to expand to 100 kilobits per second, a recording computer would require a bandwidth of 3 to 4 times this amount to handle all storage and retrieval operations in addition to the communications tasks.

In addition to the bandwidth requirements, the recording computer and its peripherals must be reliable as a system. The hardware and software must have good recovery from device errors and tolerance of some downtime for each peripheral device.

There are advantages to using the Pluribus in recording. It is already performing the data acquisition (CCP) functions. It has both the high bandwidth and high reliability requirements. Furthermore, by adding memory and processors the Pluribus can accommodate small to large expansions in the input stream. Finally the use of the Pluribus would eliminate the cost of a separate recording computer at the SDAC when the current recording computer (IBM 360/40) is retired, a move that may be necessary if the input data stream increases appreciably.

On the other hand, there are disadvantages to using the Pluribus as a recording system. The chief disadvantage is the lack of a file management system which would be necessary to support disks and tapes. Since the Pluribus was intended as a communications processor, the designers never saw the need of generating a file management system. The development costs of the file management system plus the applications software for a seismic network which may change configuration and inputs from time to time could be excessive. In fact these software costs would probably exceed the cost of an

extra computer system for the recording functions.

Another disadvantage is that mixing the communications and the recording functions in the Pluribus would make a complex system even more complex. The problems the SDAC has had up to now--especially concerning the incomplete documentation, a system not widely known among systems programmers, and the dependence upon a few key employees who know the Pluribus and the applications--would be amplified. The system would become harder to learn, more susceptible to software errors, and more dependent upon critical employees.

We do not recommend adding data recording to the CCP functions if the hardware for the job is to be the Pluribus. The software development (see Table 6.1) and the software maintenance costs would be excessive, especially considering the likelihood that the applications software would change occasionally as the seismic network configuration changed.

TABLE 6.1

	Software Modules	Development Time
1.	Overall specifications and design	3 man months
2.	Protocol module for interfacing CCP file system to external computers (NEP).	6 man months
3.	Archival module (disk to tape)	2 man months
4.	Input/output modules (collecting station buffers for recording and retrieval.	2 man months
5.	Access method modules (QSAM and random).	4 man months
6.	Device drivers for disks and tapes (read/write primitives, error recovery primitives).	6 man months
7.	Integration into current STAGE/ CONFIG modules.	2 man months
8.	Acquisition or development of more extensive system development tools (e.g. cross compilers).	6 man months
9.	Documentation & training	9 man months
	TOTAL	40 man months

Software development tasks to install data recording with CCP functions on the Pluribus. These estimates are conservative not reflecting the R&D nature of the seismic network with its likelihood of changing design specifications, requirements and seismic network configuration.

6.2 Signal Processing in the CCP

The addition of signal processing to the CCP functions in the Pluribus would be feasible since these functions would not require the use of disks or tapes, but considering the difficulty of programming the Pluribus, the R&D nature of the seismic signal processing, and the likely variations in the seismic network, such additions to the Pluribus applications software are not recommended.

Additional storage capacity and bandwidth would be needed to add signal processing to the CCP functions in the Pluribus. Signal processing algorithms must have access to the time series data. Although all data comes through the Pluribus for the CCP functions, the signal processing algorithms could require a much longer time window of the data held in memory than that needed for the communications functions. Plans for signal processing should not be restricted to recursive algorithms on single data channels. Allowance must be made to include beamforming, spectral computations, and phaseless filters if desired. All of these processes require more than a single second block of short period seismic data. Hence, the memory requirements would exceed what the Pluribus currently possesses by several times.

Other requirements include a modular structure for the signal processing software. Basic functions such as filtering, beaming, de-gainranging, demultiplexing, and detection algorithms would be applied to channels from every station in the seismic network. For R&D purposes, multiple detection algorithms should be possible on the same channel(s).

Still another requirement is a means of announcing a detection. A signal arrival queue (SAQ) would be formed in the CCP and passed onto the recording computer along with the data. The analysis system (NEP) ultimately would retrieve this SAQ from the recording computer as the analysis session starts on that time block of data. If the detection list (SAQ) must be displayed as signals are detected, then a console printout must be generated either by the CCP or the recording computer.

The advantages of using the Pluribus for both the CCP functions and the signal processing (detection functions) are bandwidth and reliability. The bandwidth of the Pluribus is adequate to handle the extra processing functions and the machine is modularly expandable in numbers of buses and processors to accommodate the modular increases in the seismic network inputs.

The reliability of the Pluribus makes it attractive to put all detection processing as well as the communication functions on line. In this way the detection list (SAQ) is always up-to-date, regardless of problems with the analysis computers.

The disadvantages of using the Pluribus for both the CCP and signal processing functions arise mainly from the difficulty of programming the Pluribus. With excess bandwidth in the computer responsible for detection processing, running various types of detection algorithms in parallel is attractive. For example local earthquakes could be flagged as such with approximate locations given by algorithms utilizing high frequency content, rotation of horizontal components, and recognition of P and S phases. For the development and testing of such algorithms a higher level language should be available. Such is not the case for the Pluribus. These signal processing algorithms would be off-line on another computer, of course. However, imbedding this signal processing code in the communications code in the CCP would not be a trivial task. Table 6.2 gives our coarse estimates for generating this software. Generally, such mixing of signal and communications codes will make a complex system even more complex, giving rise to similar software costs and maintenance problems anticipated by mixing CCP and recording functions discussed in the last section. Certainly such a software environment would inhibit experimenting with the signal processing.

Another aspect of this same disadvantage is that mixing CCP and signal processing functions in the Pluribus would be mixing non-critical functions (signal processing) with critical functions (communications and data acquisition). Certainly signal processing functions are important but they can be done off-line, close to real-time. The CCP functions, however, must be done on-line. By mixing the two, we run the risk of degrading the system reliability unnecessarily with software errors.

TABLE 6.2

Previous SDAC effort:	
LASA/DP reconfiguration (fixed algorithms; no flexibility)	4 man years
Estimate of new CCP programming	
Convert 360 DP to Pluribus	4 man years
Revise DP for greater modularity and R&D flexibility	2 man years
TOTAL	6 man years

Coarse estimate of software development effort to provide signal detection processing and CCP function on the Pluribus.

6.3 Local Network

While not eliminating the need for CCP functions, a local area network could connect the SDAC computers responsible for data recording, detection processing, event analysis, data storage and retrieval, software development, and seismic R&D while still allowing for easy transferring of data, programs, computational loads, and modular expansion.

Whenever more than two computers in one facility are connected together in some fashion, we can refer to the ensemble as a local network. In this section we want to discuss whether all of the computers, present and future, at the SDAC should be connected into a local network, and especially into a network which utilizes the network protocols which have been internationally agreed upon as standards. Further, we want to discuss whether the Pluribus or some other computer should be the hub of such a local network at the SDAC.

Digital data will be transmitted by packet-switched networks almost exclusively in the near future. Packet switching, pioneered by the ARPANET (Roberts, 1978) has been used at SDAC for data links to NORSAR and the mass store for several years. Commercial networks are available (e.g. Tymnet, Telenet, Cybernet, Bell Canada) offering reliable services with the overhead software transparent to the user. In 1976 the International Telegraph and Telephone Consultive Committee (CCITT) approved standard protocols (x.25) for interface host computers and data terminals. That same year four packet carriers--Telenet in the USA, Bell Canada, and the public packet-switching carriers in the United Kingdom and France--adopted these same standards. IBM, DEC, Honeywell, Data General, and, most other manufacturers of minicomputers or terminals are supporting these standards. Any future expansion of the world-wide seismic network could well utilize one or more of these commercial networks in its data acquisition.

Because these packet-switching protocols have become standardized and supported by the minicomputer and terminal manufacturers, there may be advantages for a seismic network to extend packet-switching into its central facility (the SDAC for the VELA network) by way of a local network. Data could be passed from CCP to the DP, NEP, mass store, SDAC storage/retrieval, scientific processing, and software development computers at SDAC or on the ARPANET using network control software already developed.

However, such a revision at SDAC must consider some differences between long-haul networks and local area networks. A long-haul network must be able to connect any node of many to any other, while the local network has fewer expected connections among far fewer nodes. The long-haul network must expect wide variations in traffic, while the local network often can expect steadier and more deterministic traffic. The long-haul network must charge for distance and usage so bandwidth is limited and expensive; the local network installs its own links, so bandwidth is abundant and cheap.

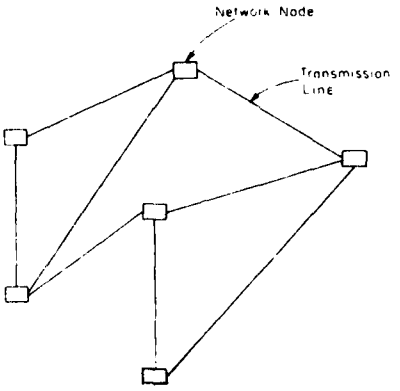


Figure 6.3a Unconstrained topology. Each node receiving a message must make a routing decision to forward the packet to its final destination.

As a result of these differences, local networks usually exist in simpler topologies and use simpler protocols. Long-haul networks will use unconstrained topologies such as that illustrated in Figure 6.3a. The advantage of the general topology is that the communication links can be based upon network traffic and thus optimize the use of costly transmission media. This arrangement complicates protocols, since each node must make a routing decision for each arriving packet. For a node to blindly transmit all messages not for itself to every outgoing link would multiply message traffic at each node and keep some messages propagating forever.

On the other hand, local networks frequently use one of three much simpler topologies: the star, the ring, and the bus as shown in Figure 6.3b. Even with these simpler structures protocols can become involved. For a ring, one simple protocol would have all messages pass counterclockwise, and each node submitting a message would be responsible for removing it from the ring. The assumption is that the destination node has received it and no formal acknowledgement is sent. This structure requires all nodes to stay alive or the ring is broken. The simplest protocols occur in the star since only the center node needs to make routing decisions. This topology is ideal for a primary node, for instance a time-sharing scientific computer, communicating with a number of secondary terminals. However, if two of the nodes handle most of the traffic between them, then the reliable operation of this dominant traffic hinges on the reliability of its central hub in addition to the reliability of the two nodes.

For the SDAC, a feasible local network could be established using the Tandem computer discussed in the previous section. The Tandem machine is similar to the Pluribus in architecture and reliability. However, it has

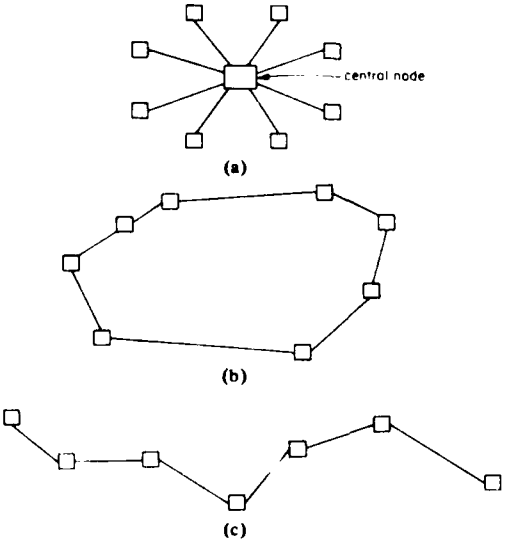


Figure 6.3b Examples of constrained topologies. (a) The star. (b) The ring. (c) The bus.

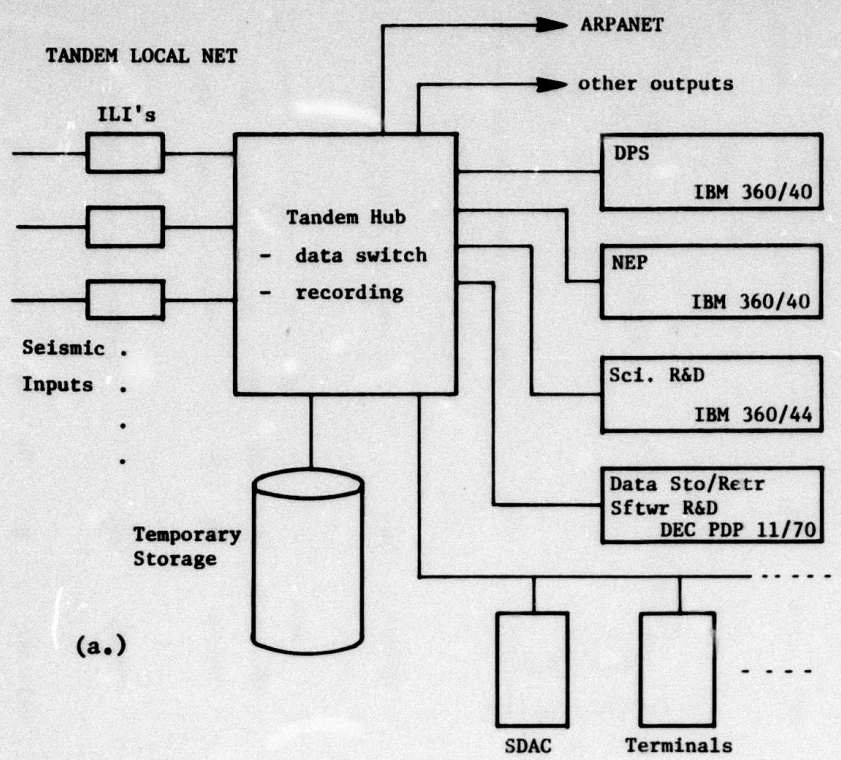
some advantages which make it ideal for the central hub of a star local network. It can act as a gateway between the local network and the long-haul network for data acquisition and distribution (i.e. CCP functions). It supports disks and tapes, so it can be a recording machine. Preventive maintenance and development work can be done on part of the machine off-line without disturbing its on-line functions. It supports FORTRAN and all of the international standard (CCITT) protocols (X.25, X.28, X.29, and X.3) adopted by the long-haul packet-switched networks. Hence, the software development for an SDAC would be as simple as possible with today's technology. The other SDAC machines responsible for detection (DPS), analysis (NEP), storage and retrieval, scientific R&D, and software development would be connected to it as off-line satellites or secondary nodes. (See Figure 6.3b (a)).

A similar network could be arranged for the SDAC using the Pluribus. In this case, the Pluribus would probably not be responsible for recording but funnel all data to a minicomputer for that purpose. Two variations are worth considering. In the first (Figure 6.4a), the topology would match that of the Tandem local network with the recording computer as a secondary node, but the only secondary node which is, or must be kept, on-line. The Pluribus would be the data switch at the hub. In the second, (Figure 6.4b) the topology would place the Pluribus as CCP (plus the Intelligent Line Interfaces, ILI's, if any) and its recording computer as the only on-line spoke of the star. The recording computer would be connected to another machine, such as a small Tandem, at the hub. The hub and all other nodes (DPS, NEP, scientific R&D, etc.) would be off-line. In each case the CCP functions do not disappear with a local network, but must be retained as a gateway between the long-haul networks used for data acquisition and distribution and the local network used for seismic recording, analysis, and R&D.

We might speculate here on the potential seismic impact of these computer network developments. As the world-wide seismic network expands in numbers of sites, the seismic data volume arriving at SDAC, or some other national data center, will increase. If the waveforms and analyses of all recent events from key areas of the world are saved in a mass store, together with historic events of known explosions and earthquakes, then the traffic into and data edited out of the mass store will increase.

How much data should be saved for each event, and how many events should be saved on-line in a mass store is debatable. Even with minimum intervals per event the seismic data traffic could exceed the capacity of the ARPANET. If so, a new and probably larger mass store would be established at the seismic data center (SDAC or other) some time in the future. More automated procedures for editing, storing, and retrieving such data would demand a local area network to provide easy access to the data bank from the computers involved with data acquisition, analysis, and R&D. If computer terminals at the SDAC are programmed to recall, sort, and analyze event bulletins and also to recall, display, and compare recent event waveforms with those from historic events, then the demand could arise in government agencies, universities, and data centers in other countries for similar automated access to the seismic data bank. Ultimately the mass store technology will improve so that 100% of the seismic network data from several decades can be kept on-line (e.g. mass store of 10^{15} bits in 10 years or so). Then universities and data centers all over the world will want automated access to the seismic data bank. The way to most easily accommodate advances in computer technology of the future may well be to establish a local area network at the SDAC based upon the international standard (CCITT) for packet-switched data communications.

More study will be required to assess the advantages of a local network for the seismic system before a particular topology and system could be recommended. Just how much software development would be required for the Pluribus and for the Tandem local networks, or some other one, we cannot say at this time. We do not know how such systems would affect the seismic data base and its management system (DBMS).


TANDEM LOCAL NET
ILI's
ARPANET
other outputs
Tandem Hub
- data switch
- recording
DPS
IBM 360/40
NEP
IBM 360/40
Sci. R&D
IBM 360/44
Data Sto/Retr
Sftwr R&D
DEC PDP 11/70
Seismic
Inputs
Temporary
Storage
(a.)
SDAC
Terminals

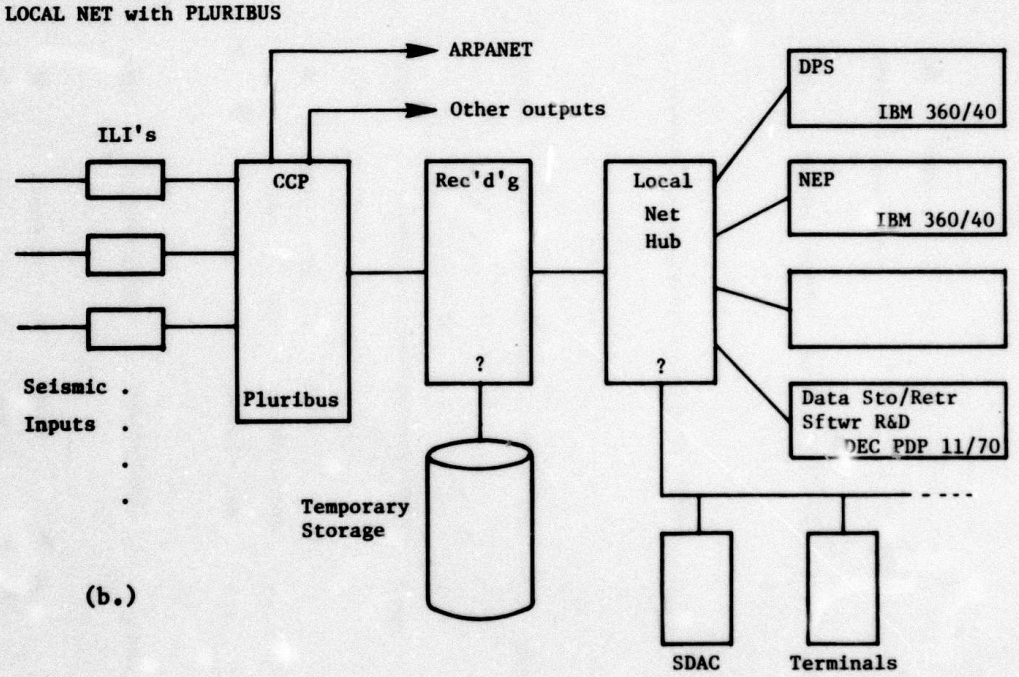

LOCAL NET with PLURIBUS
ARPANET
Other outputs
ILI's
CCP
Rec'd'g
Local
Net
Hub
DPS
IBM 360/40
NEP
IBM 360/40
Data Sto/Retr
Sftwr R&D
DEC PDP 11/70
Seismic
Inputs
Pluribus
Temporary
Storage
(b.)
SDAC
Terminals

Figure 6.4

This Page
Intentionally
Left Blank.

REFERENCES

REFERENCES

1. Baskin, M. B., Borgerson, B. R., & Roberts, R.; "PRIME - A Modular Architecture for Terminal Oriented Systems," Proc. SJCC, Vol. 40, pp. 431-437, AFIPS Press, Montvale, N. J. 1972.

2. Bolt, Beranek & Newman Inc. staff, Pluribus Document #2: System Handbook, BBN Report No. 2930, Jan. 1975, pg. 4.

3. Briscoe, H., VELANET Communications Protocol between the CCP and SIP and between CCP and DP, BBN Report No. 3460, 5 April 1977.

4. CCITT, Provisional Recommendations x.3, x.25, x.28 and x.29 on packet-switched data transmission services, the International Telegraph & Telephone Consultive Committee, ISBN 92-61-00591-8, Geneva 1978.

5. Clark, D. D., Porgran, K. T., and Reed, D. P., An Introduction to Local Area Networks, Proc IEE, 66(11), Nov. 1978, pp. 1497-1516.

6. DataPro Reports on minicomputer, 1979, DataPro Res. Corp., Delran, N.J. 08075, McGraw Hill.

7. Katsuki,D., Elsam, E. S., Mann, W. F., Roberts, E. S., Robinson, J. G., Skowronski, F. S., and Wolf, E. W., Pluribus-An Operational Fault-Tolerant Multiprocessor, Proc. IEEE, 66(10), Oct. 1978, pp. 1146-1159.

8. Losq, Jacques; "Effects of Failures on Performance of Gracefully Degradable Systems," Tech. Note #103, Digital Systems Lab, Stanford University, Dec. 1976.

9. Roberts, L. G., the Evolution of Packet Switching, Proc. IEEE, 66(11), Nov. 1978, pp. 1307-1313.

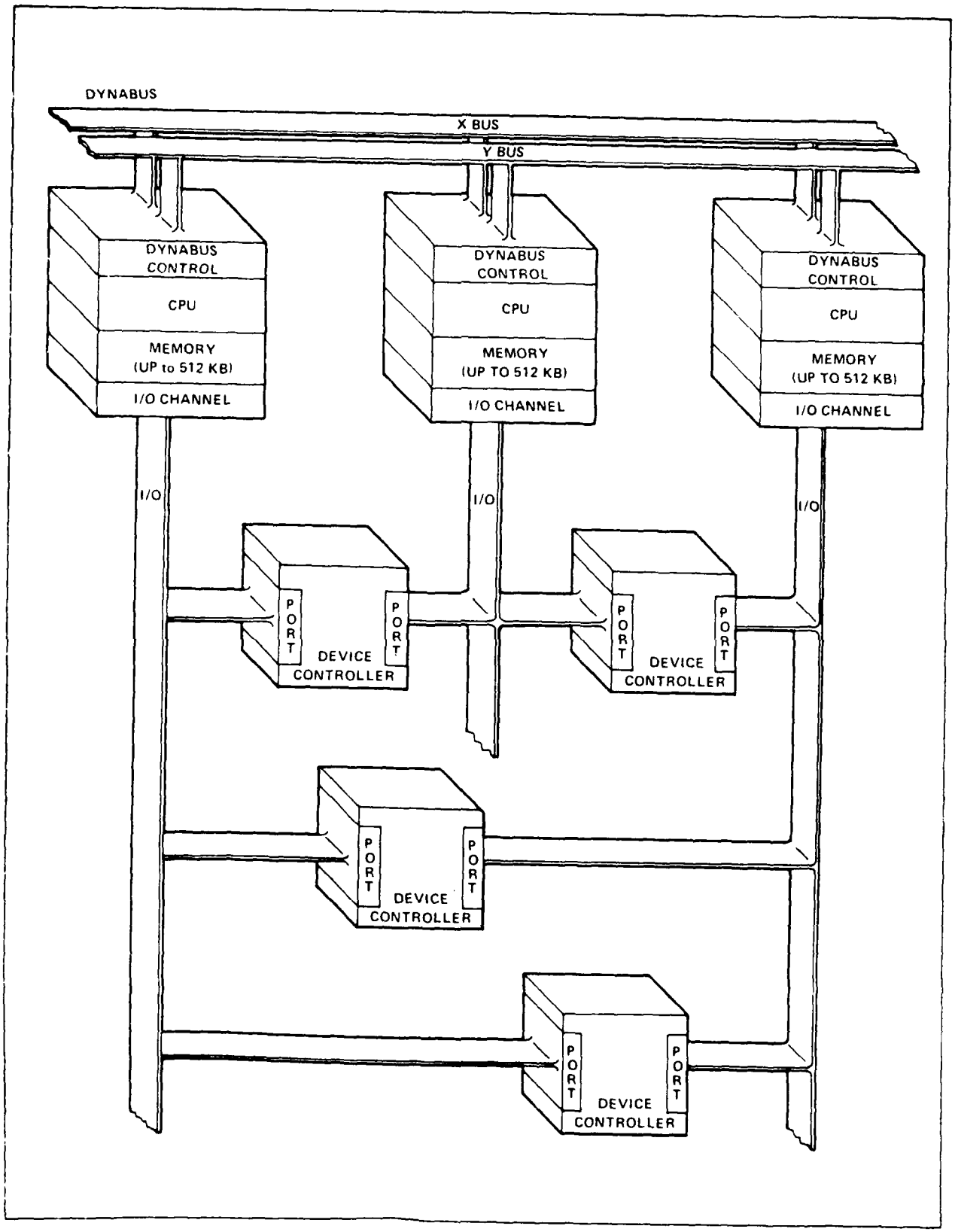
10. Texas Instruments staff, Specification for LP Darts station controller to HDT communication formats, Code Ident. Nos. 96214 and 18667 Sept. 1975.

APPENDIX I

Tandem Non-Stop Systems

NonStop™ SYSTEMS

INTRODUCTION



Tandem 16 Computer System

The Tandem 16 Computer System fills three important and interrelated needs: it is a computer system where applications run *NonStop* regardless of a module failure, it provides a computer system that can support high transaction rates to large on-line data bases, and it provides a system that is easily adaptable to any application.

The basic design philosophy of the Tandem 16 Computer System is that no single module failure will stop or contaminate the system. This assurance of *NonStop* operation is sometimes called "fail-safe" when no loss of throughput occurs as a result of a failure or "fail-soft" when some slowdown occurs but full processing capabilities are maintained. The Tandem 16 can provide both "fail-safe" and "fail-soft' modes of operation.

- **PROCESSOR MODULES**

  A single Tandem 16 Computer System may contain from two to sixteen processor modules; each module contains a micro-programmed central processing unit, its own memory (up to 512k bytes), and its own micro-programmed input/output channel. Each processor module is fully capable of operating independently of all other processor modules yet can be configured to back up other processor modules.

- **INTERPROCESSOR BUSES (DYNABUS)**

  Each processor module is connected to all other processor modules via redundant high speed interprocessor buses. Programs running in one processor module communicate with programs running in other processor modules by means of these buses. Each interprocessor bus is fully autonomous, operating independently of (but simultaneously with) the other bus. The use of two buses assures that two paths exist between all processor modules in the system.

- **INPUT/OUTPUT CHANNEL**

  Input/output devices (i.e, magnetic tape units, disc drives, terminals, etc.) are interfaced to the computer system via dual-port i/o controllers. Dual-port means that each i/o controller is connected to the input/output channels of two processor modules. This provides two paths of communication to each input/output device. A dual-port controller is "owned" by (i.e., will accept commands from) only one processor module. But in case of a failure, the other processor module can take control programmatically. The dual-port controllers are designed so that the number of components common to both paths are at a minimum.

- **TANDEM 16/TRANSACTION OPERATING SYSTEM (T/TOS)**

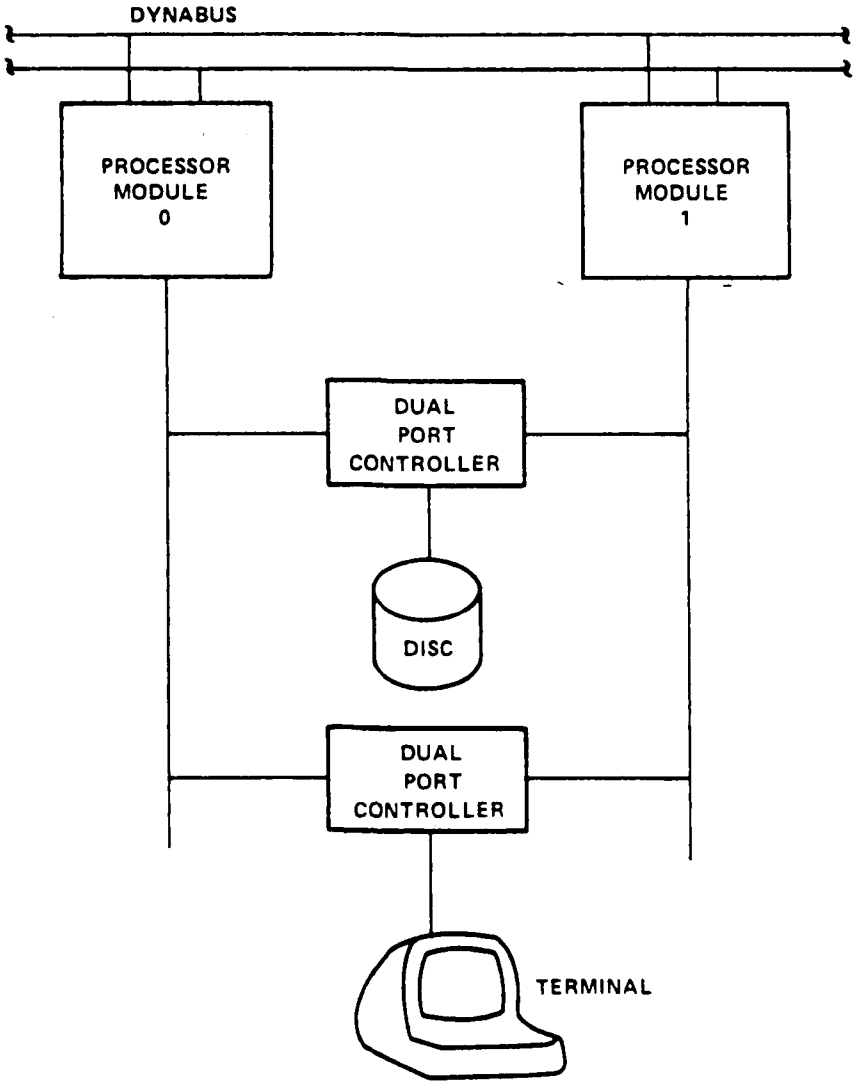
  Overseeing system operation is the Tandem 16 Operating System. The operating system provides the multiprocessing (parallel processing in separate processor modules), multiprogramming (interleaved processing in one processor module), and *NonStop* capabilities of the Tandem 16 Computer System. A copy of T/TOS resides in each processor module.

The operating system automatically schedules application programs for execution according to an application-assigned priority, provides memory management functions (automatic overlaying, swapping to disc, etc.), and gives application programs the capability to start programs executing in any processor module from any processor module.

- **FILE SYSTEM**

As part of the operating system, the Tandem 16 File System provides a single interface between programs and the outside world. The file system, which handles all data transfers, provides the capability for any program running in the system to communicate with any other program running in the system as well as any input/output device connected to the system; programmers need not be aware of the physical location of the device/program. Up to 4,094 bytes can be transferred in one file system operation.

A minimum TANDEM 16 NonStop™ System:



SUMMARY OF TANDEM 16 FEATURES

Tandem 16 Computer System (fail-safe, transaction oriented, easily adaptable)

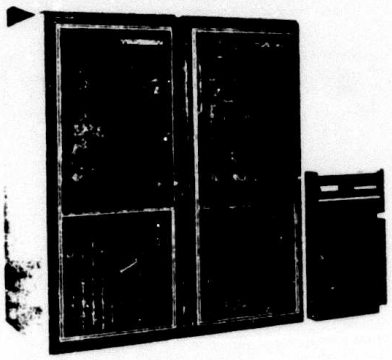
- Two to sixteen processor modules
- Dual, high-speed interprocessor buses
- High-speed, burst-multiplexed input/output channel
- Multiprocessing, multiprogramming, transaction oriented, fail-safe operating system (T/TOS)
- Virtual file system

Processor Modules

- Microprogrammed (100 nanosecond cycle time)
- Sixteen bit data paths and memory addressing
- Up to 512k bytes memory per processor module
- Memory mapping (four separate maps: system data, system code, user data, user code)
- Up to 128k bytes addressable through each map (for a total of 512k bytes addressable)
- 500 nanosecond semiconductor memory access time (including mapping and error correction)
- 800 nanosecond core memory access time (including mapping and parity checking)
- 123 instructions (including string manipulation and doubleword arithmetic)
- Four-Word (QUAD) Decimal Option: 20 additional instructions (including multiply, divide, rounding, convert quad value to ASCII, convert ASCII to quad value)
- Stack architecture (memory stack and register stack)
- Procedure oriented hardware
- Memory references: direct or indirect with or without indexing to global, local, procedure parameters, top of stack, or system global data areas and to code area. Word, doubleword, byte, and four-word, addressing
- No machine instruction can write into code area (non-modifiable code)
- All programs are inherently re-entrant and relocatable
- I/O transfer, bus receive, and instruction execution occur concurrently
- Next instruction prefetched while current instruction executes
- Hardware power fail/auto restart
- Hardware multiply/divide
- Maximum of five microseconds to call operating system procedures

Interprocessor Buses

- Two paths between each processor module
- 10 megabyte transfer for each bus



TANDEM

NonStop™ STANDARD PACKAGED SYSTEMS

- **NonStop™ Multi-Processor Systems for On-Line Transaction-Oriented Applications**
- **Autonomous/Dual DYNABUS™ for Inter-Processor Communications (13 Mega Bytes/second each)**
- **Fast (800 or 500 nSec) Core or Semiconductor Memory with Virtual Memory Control and Automatic/Dynamic Program Allocation**
- **High-Speed Redundant Dual-Port I/O Channels with Dedicated Microprocessed Interrupt and Block-Multiplexed DMA (2.5 or 4.0 Mega Bytes/Second)**
- **Versatile System Applications Support with Tandem/Guardian and Tandem/Transaction Applications Language (T/TAL)**
- **100 ns cycle time Processors**

SYSTEM CONFIGURATIONS

The Tandem NonStop systems (illustrated on the reverse side of this page) are packaged especially for critical on-line transaction-oriented applications where high reliability and low program overhead are essential. All systems feature dual processors with Tandem's unique DYNABUS for high-speed inter-processor communications, and redundant dual-port I/O channels for input/output resiliency. Each individual processor module contains two microprogrammed processors: one dedicated to programs and one dedicated to input/output control and data transfers. In addition, each central processor provides Power Fail/Auto Restart, an Interval Timer, Hardware Multiply/Divide, DMA, Dynamic Memory Mapping, Virtual Memory control, and a Bootstrap Loader.

TANDEM T16/212-1 SYSTEM

The Tandem T16/212-1 System consists of a dual processor package utilizing core memory modules. Each of the two processors contains 192K bytes of 800 nanosecond core memory with parity. The memory is arranged in 64K byte modules (32K words of 17 bits each). The T16/212-1 may be easily expanded in the field to 448K bytes of memory on each processor. The T16/212-1 I/O System provides high speed dual-port I/O Channels with block-multiplexed DMA. The data rate of each I/O Channel is 2.5 MBytes/second. The system cabinet provides 14 vacant slots for expansion of I/O Controllers.

TANDEM T16/244-1 SYSTEM

The Tandem T16/244-1 System consists of a dual processor package utilizing semiconductor (MOS) memory modules. Each of the two processors contain 192K bytes of 500 nanosecond semiconductor memory with error detection and correction. The memory is arranged in 96K byte modules (48K words of 22 bits each, 16 for data, 6 for ERCC). The T16/244-1 may easily be expanded in the field to contain two additional processors for a total of four processors and each processor may be expanded to 512K bytes of memory. Battery backup is supplied as a standard feature on semiconductor memory. The T16/244-1 I/O System provides high speed dual-port I/O Channels with block multiplexed DMA. The data rate of each I/O Channel is 4.0 MBytes/second. The System Cabinet provides 14 vacant slots for expansion of I/O Controllers.

SUBSYSTEM CONFIGURATIONS

All Tandem packaged systems are supported with a wide selection of peripheral subsystems to meet the heavy demands of diversified applications and high-volume data bases.

Disc Subsystems – Because disc requirements are highly application dependent the user will add disc controllers and drives as needed. The user can choose from a wide variety of disc controllers and drives including 10MB, 50MB and

160MB capacities. Through the use of appropriate controllers and software the user can configure redundant "mirrored" data bases for increased system integrity.

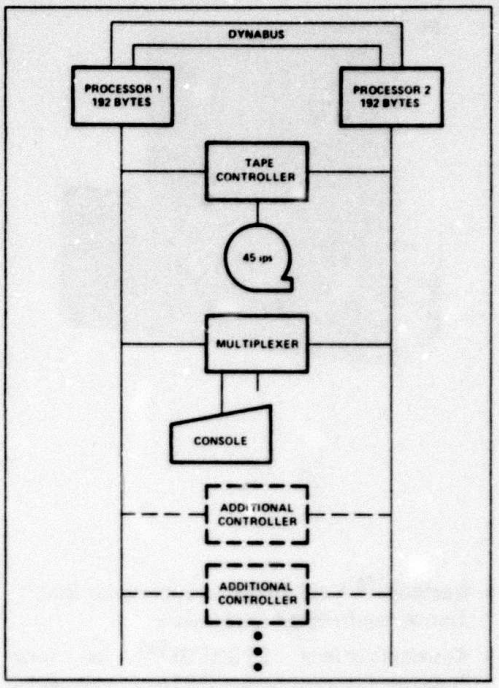
Tape Subsystems – A Magnetic Tape Drive and Controller make up the Tape Subsystem. The tape unit is a 45 ips, 9-track, 800/1600 Dual Density NRZI/PE BPI drive with read-after-write and power-fail/auto-restart electronics. The dual-port controller can handle up to two independently connected tape drives with no daisy chaining required. The Controller may be powered from either one of the dual processors.

Terminal Subsystem – The basic Terminal Subsystem consists of a Multiplexer and the system Console. The Multiplexer is comprised of an Asynchronous Controller. The dual-port controller provides two independent 50 to 19.2K baud communication lines for modem or direct-wire connection compatible with single- or multi-drop terminals. One line is dedicated to the system's 30 CPS, 132-column hard-copy console.

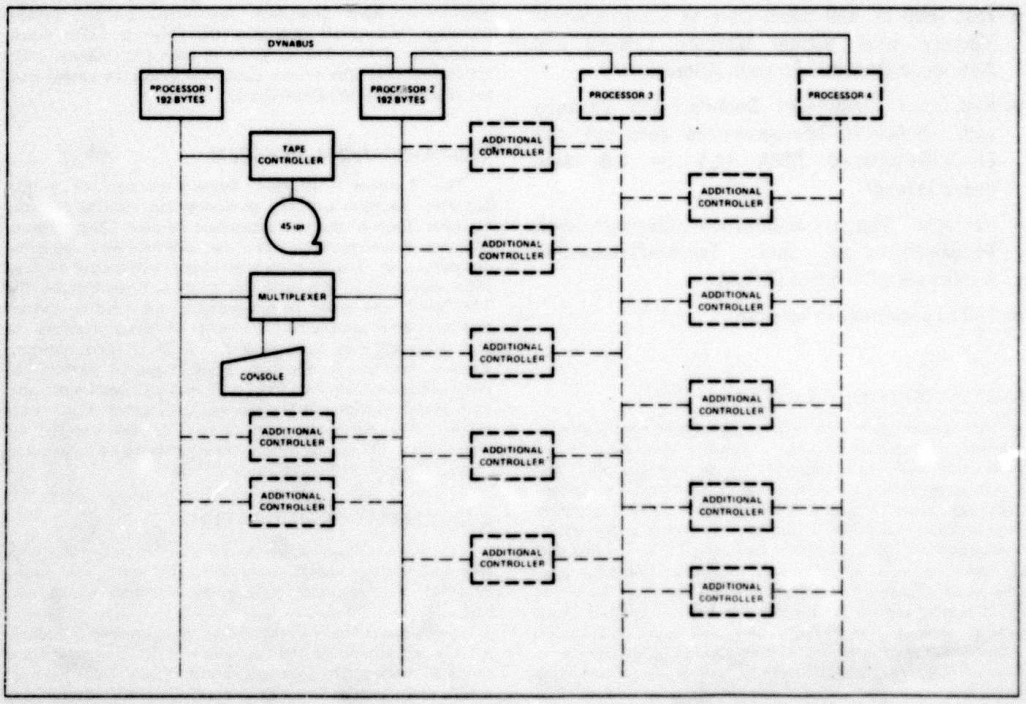
APPLICATIONS SUPPORT

Tandem packaged Non-Stop systems come with full applications program development support: the Tandem/GUARDIAN Operating System, Source File Editor (EDIT), Object File Editor (UPDATE), and an interactive Debugger (DEBUG). The Tandem/Transaction Applications Language (T/TAL) compiler allows programs to be written for NonStop applications in a high level language. This provides self documenting programs and eliminates the time consuming errors inherent in assembly language. In addition, utilities are provided for backup/restore, file creation, system generation, and system operation.

SYSTEMS T16/212-1



SYSTEMS T16/244-1



TANDEM

GUARDIAN OPERATING SYSTEM

FEATURES

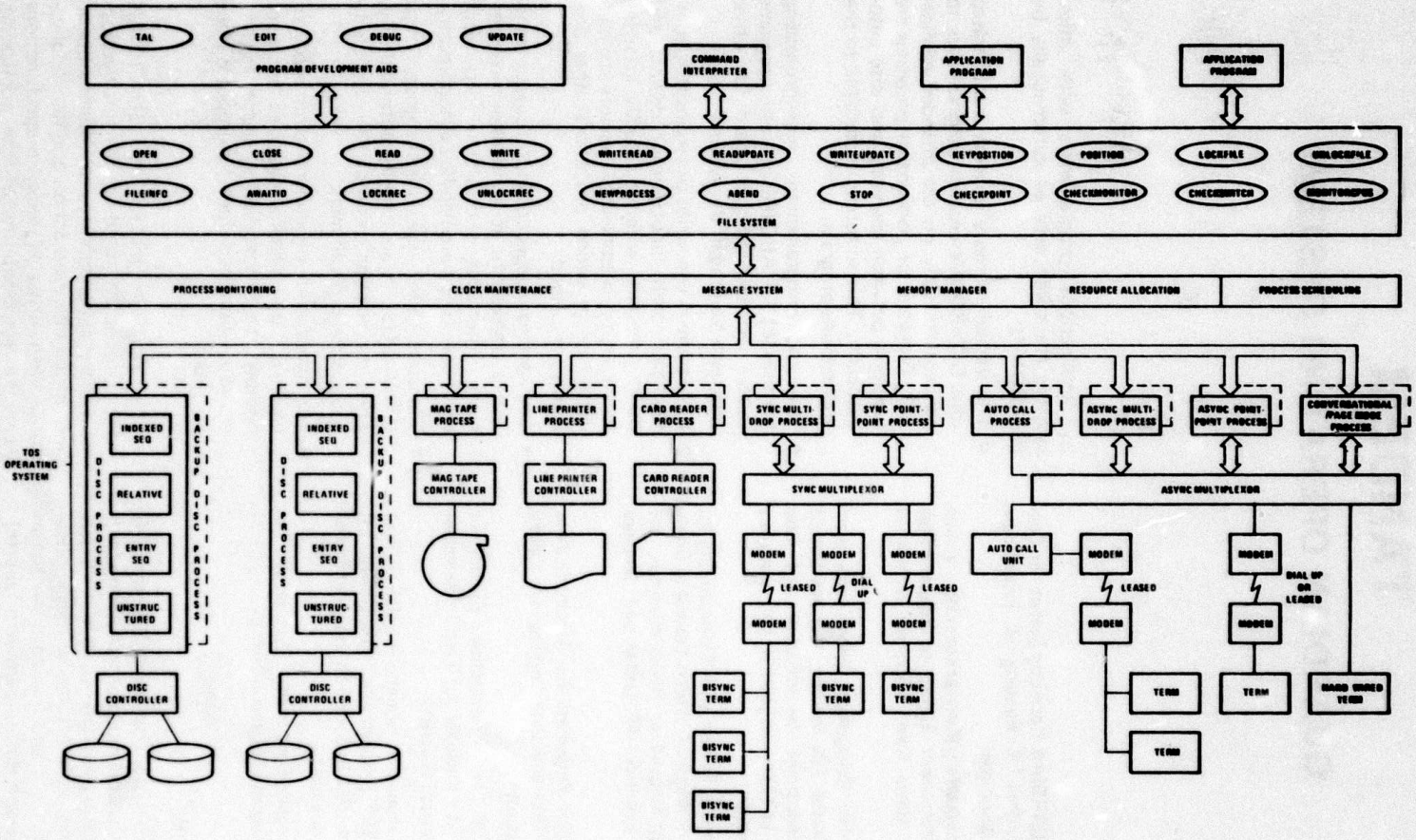
- **Fail-Safe GUARDIAN Operating System for Continuous Non-Stop™ Running of Transaction-Oriented Applications**
- **Multiprogramming/Multiprocessing Virtual Memory Management System to Support High Transaction Rates from Numerous High-Speed Terminals**
- **Redundant File Management System with Symbolic File Access, File Security, File and Record Locking, Concurrent Input/Output and Disc Volume Interchangeability without Reprogramming**
- **Fail-Safe Message System with Checkpointing for Fault-Tolerant Programs, Process Control to Establish, Change, Suspend or Delete Processes, and Automatic Resource Allocation and Memory Mapping**
- **Fail-Safe Utility Procedures for Automatic Data Conversion, Time and Date Logging, and Calling the Debug Facility**
- **Efficient High-Level Transaction Application Language (TAL) Compiler for Easy Implementation of Application Programs**
- **Interactive Command Interpreter (COMINT), Text File Editor (EDIT), Object File Editor (UPDATE), and Debugger (DEBUG) for Fast and Economical Program Development**
- **On-Line Program-Concurrent Diagnostics for System Security and Program Integrity.**

INTRODUCTION

The GUARDIAN Operating System is a true multi-processor, multi-programming *fault tolerant* operating system. GUARDIAN has been specifically architectured so that applications can be defined and implemented to run continuously (even during hardware failures). Many of the responsibilities normally handled by applications programmers with other systems are taken care of automatically by GUARDIAN:

- The virtual memory scheme incorporated into GUARDIAN enables programmers to concentrate fully on the intended application. Programs or portions of programs are swapped between memory and disc automatically. This swapping is invisible to the applications program.
- The multiprogramming, multiprocessing features of GUARDIAN permit programs to be written without regard for other programs running in the system.
- Programmers can write programs that communicate with input/output devices and other programs without actually knowing to which processors these devices are connected or in which processors other programs are located.
- The hardware aspect of input/output transfers is transparent to application programmers. Interrupts and system dependent error conditions are handled automatically.
- Transfers over the interprocessor bus (DYNABUS)™ are handled automatically. The bus operation is completely invisible to application programmers.
- For application dependent error recovery routines, the system provides an error number that specifically describes any errors encountered during an input/output operation.

The *Fail-Safe* features of GUARDIAN make it uniquely suited for the fast development, easy expandability, and reliable operation of on-line transaction-oriented applications. With other systems, these functions were the responsibility of the application programmer.



Guardian Operating System

VIRTUAL MEMORY MANAGEMENT

Within each processor the operating system is responsible for allocation of execution time to multiple programs on a priority basis; allocation of buffer space and control blocks; process synchronization; fault and trap handling; and interval clock maintenance. In addition, GUARDIAN isolates the application from physical memory constraints by providing an efficient virtual memory management system. Paging hardware is provided in the form of four 128K byte memory maps (user code user data, system code and system data). All code is both sharable by multiple programs and non-modifiable, two features which reduce overlay and swapping overhead. In addition, hardware is used to record frequency of access to all memory pages and modification of data pages, thus providing a low overhead method of determining the correct page to be replaced.

FILE SYSTEM

Provided as part of GUARDIAN, the *File Manager* is a flexible, easy-to-use device-independent interface. It allows an application program to communicate with disc files, serial I/O devices, conversational page mode and multidrop terminals, and other application programs all through one standard set of routines. The File Manager allows program execution to continue concurrent with execution of the file request. In fact, the input/output may be taking place in a different processor module.

The File Manager also includes a number of features unique to the *NonStop* environment. When a processor or I/O port fails, the File Manager automatically reroutes subsequent requests to a back-up processor module. In addition, through a very efficient program-to-program communication mechanism, the File Manager provides the application with a simple method of keeping back-up programs informed of current operations so that a smooth transition may be made.

File Access – For I/O devices normally dedicated to a single process, such as Terminals or Line Printers, the device is assigned a *symbolic* name. The programmer need not know the physical address of the device. This provides a simple means to add and/or reconfigure I/O devices without reprogramming the application. In the case of disc devices, a file name represents a user-specified portion of the disc storage space. The File Manager makes no distinction between sequential and random access to a disc file. A file pointer (relative byte address) determines where a data transfer is to begin. The file pointer is normally automatically incremented for sequential access but may also be set explicitly for random access. An access mode specifies the operations to be performed on a file:

- **Read/Write (Default Mode)**
- **Read Only**
- **Write Only**

File Procedures – To implement file operations, calls are made to file management procedures. All files are accessed through the same set of procedures which provides a single uniform access method. These procedures include:

• **CREATE**	a new file
• **OPEN**	a file for access
• **READ**	data from a file
• **WRITE**	data to a file
• **WRITEREAD**	write/read data to/from a file
• **READUPDATE**	read data from a file for subsequent update
• **WRITEUPDATE**	write updated data to a file
• **CLOSE**	a file to access
• **PURGE**	a file from a disc

Numerous other procedures are provided for device-dependent operations.

File Security – The versatile File Manager provides the ability to limit file access to an individual, a group of individuals, or an individual within a group. This limitation is *password* protected. Four types of file operations (read, write, execute and purge) may be separately limited to either the individual (owner) who created the file the owner's group, or any individual within the group. A program uses an *exclusion mode* to limit file access. The exclusion modes are:

- **Shared Access (Default Mode)**
- **Exclusive Access**
- **Protected Access**

File *locking* is provided so that cooperative application programs may share file operations as a disc file. Also, disc volumes may be removed and replaced without reprogramming the application and without loss of file security.

MESSAGE MANAGEMENT SYSTEM

The GUARDIAN message system handles all communications between Tandem 16 processor modules, system processes and application programs. It frees the user of the responsibility of routing messages to the correct processor, verifying that it got there correctly, and deciding which program is to receive it in the destination processor. A program need not be aware of which of the 16 Tandem processors will ultimately run it. In fact, the same program may be executing simultaneously on all processors. In addition, a program may access any device on the system, even if the device is not physically connected to the processor in which the program is running. This allows the system to be expanded without reprogramming the application.

Process Control – A *process* is the execution of a program under control of the GUARDIAN Operating System. Process control procedures are used to interactively call processes. These processes are:

- **NEWPROCESS** – create a new process (run a program)
- **DELAY** – suspend the calling process for a specified interval of time
- **PRIORITY** – change the process execution priority
- **STOP** – delete a process with a normal indication
- **ABEND** – delete a process with an abnormal indication

For example, to have a process delete itself and close its files, the procedure CALL STOP is used.

System Messages – GUARDIAN sends messages directly to application processes to inform the application of certain system conditions. Some of these messages are:

- **Processor Module Failed**
- **Process Stopped Execution**
- **Processor Module Reloaded**

Certain critical error conditions prevent normal execution of a process. These errors cause traps to GUARDIAN Trap Handlers.

Checkpointing – Fail-safe, non-stop operating environments require one or more primary/backup *process pairs*. The primary process executes in one processor while the backup process monitors in another processor. With this type structure, the backup process is kept informed of the primary's execution state of the primary process via periodic *checkpoint* messages sent to the backup process. Each process in a process pair has the same set of files open. This ensures that the backup process has immediate access to the files in the event of a primary processor's failure. GUARDIAN provides several procedures to aid in the development of fault-tolerant programs:

- **CHECKOPEN** –
  is called by a primary process to open a file in its backup process
- **CHECKPOINT** –
  is called by a primary process to checkpoint its current state to its backup process
- **CHECKMONITOR** –
  is called by a backup process to monitor its primary and take appropriate action in the event of the primary's failure
- **CHECKSWITCH** –
  is called by a primary process to switch control to its backup process
- **CHECKCLOSE** –
  is called by a primary process to close a file in its backup process

COMMAND INTERPRETER (COMINT)

Tandem's COMINT is a high-level man-machine interface which allows the user to converse with the system. The user may obtain or alter the current operational status of the system; create, verify and purge disc files; and run programs (both Tandem-supplied and application programs). Normally, the user initially executes COMINT on a system console. From this point on, COMINT may be specified to be run on any other terminals connected to the system.

PROGRAM DEVELOPMENT AIDS

Included as part of the standard software provided with every Tandem 16 system is a comprehensive set of program development tools. These programs, which run under control of the Command Interpreter, allow the user to develop application programs with a minimum of effort. Included are a high-level compiler, a source file editor, object file editor and interactive debugging facility.

Transaction Application Language (T/TAL) Tandem's *Transaction Application Language* is a high-level, block structured language designed for the easy implementation of transaction-oriented applications. T/TAL provides many high level constructs, including IF THEN, FOR, DO UNTIL, WHILE and CASE. It allows the programmer to write self-documenting programs and eliminates the time consuming errors inherent in assembly language programming. Special string manipulation operations are included to facilitate fast processing of transaction data; among them are MOVE, COMPARE and SCAN strings. While providing all these flexible features, T/TAL does not sacrifice execution efficiency. A highly optimized compiler, it produces object programs as efficient as those written in an assembly language.

Edit – The *Text Editor* is a very flexible, interactive text file editor that can be used to prepare both program source files and documentation. EDIT can be run from either conversational or page-mode terminals, and provides an additional interface to allow it to be driven by other programs.

Update – The *Object File Editor* allows the user to make changes to previously compiled programs. In addition, the output of multiple compilations can be combined through the facilities of UPDATE.

Debug – An interactive program debugging facility supported by GUARDIAN enables the user to test application programs. It provides program breakpoints, tracing of variables, and access to all of the code and data of the program, all from an interactive terminal.

TANDEM
ENVOY™
DATA COMMUNICATIONS MANAGER

FEATURES

- **Fail-Safe Data Communications Manager for Non-Stop™ Transaction-Oriented Applications**
- **Multiple Communications Protocols**
  - **IBM Binary Synchronous (BSC)**
  - **ADM-2 Asynchronous**
  - **TINET Asynchronous**
  - **Burroughs Synchronous**
- **Multiple Line Types (Data Links)**
  - **Point-to-Point (BSC only)**
  - **Centralized Multipoint Supervisor**
  - **Centralized Multipoint Tributary**
- **Trace Facility, Line Usage & Error Statistics, On-line Testing**
- **Support for Auto Call Facility**
- **Support for Multiple Modem Types**
  - **Bell type 103, 202 Asynchronous**
  - **Bell type 201, 208 Synchronous**
- **Simplified Applications Programming with Calls to GUARDIAN Operating System Procedures**

INTRODUCTION

The ENVOY *Data Communications Manager* provides an efficient, easy-to-use interface between transaction-oriented application programs and the telecommunications network. And since ENVOY is under control of the GUARDIAN *Operating System,* it has the same inherent *fail-safe* features of all other Tandem systems software. As such, ENVOY ensures that data communications are maintained even in the event of a processor or I/O channel failure. ENVOY provides all the control necessary for switched or leased point-to-point and leased multi-point telecommunication networks. An automatic calling option allows unattended stations on switched networks to communicate and exchange data without operator intervention. The ENVOY Data Communications Manager supports Synchronous transmission at speeds up to 56K Bps and Asynchronous transmission at speeds up to 19.2K Bps.

POINT-TO-POINT DATA LINK

ENVOY supports a Binary Synchronous point-to-point data link over either a switched or leased (non-switched) lines. In the switched network mode, each of two stations has a unique telephone number. The originating station dials the answering (remote) station and establishes a connection. After the connection has been established, system security messages may be interchanged to ensure the integrity of the two stations. Information interchange may then proceed. Once the interchange has been completed, the calling station automatically disconnects. The answering station detects the disconnection and then it also automatically disconnects. In the leased (non-switched) mode, the two stations are permanently connected (no number is required), but the data exchange sequence is the same as that described above for switched lines.

MULTIPOINT DATA LINK

A multipoint data link consists of two or more stations communicating over a leased (non-switched) line. With this type of data link, one station is designated the *supervisor* and controls all communications over the link. All other stations are designated *tributaries.* In a *centralized* multipoint link, communications can occur only between the supervisor and a tributary (tributary-to-tributary communication is not allowed). Each tributary station in a multipoint link is identified by a *polling* address and a *selection* address.

To solicit data transfers from the tributary stations, the supervisor station periodically *polls* each tributary station in the multipoint link by transmitting each tributary's polling address. When a tributary that has data to send is polled, further polling of the data link stops. At this point, the tributary responds by transmitting its data to the supervising station.

To transfer data to a tributary station, the supervisor station first *selects* the desired tributary station by transmitting the tributary's selection address. For selection to occur, the tributary station must be monitoring the line and be willing to accept the forthcoming data. Following selection, the supervisor station transmits the data to the tributary station.

TANDEM ENVOY™

ENVOY PROCEDURES

ENVOY functions are implemented via making calls to the GUARDIAN *Operating System's File Management Procedures*. These procedures are as follows:

- The **OPEN** Procedure is used to gain access to a data communications line.
- The **DEFINELIST** Procedure is used by application processes operating as a multi-point supervisory or tributary station. For a supervisory station, **DEFINELIST** specifies the polling address and selection address of each station. For a tributary station, **DEFINELIST** specifies the polling address(s) and selection address(s) that identify the tributary when the multi-point line is polled.
- The **WRITE** Procedure is used to transmit data to a remote station.
- The **READ** Procedure is used to accept data from a remote station.
- The **WRITEREAD** Procedure is used to transmit data to a remote station and then await a reply.
- The **HALTPOLL** Procedure is used by a station operating as a multipoint supervisor to stop continuous polling of all stations on a line.
- The **CHANGELIST** Procedure is used by application processes operating as a multipoint supervisory or tributary station. For a supervisory station, **CHANGELIST** is used to enable/disable polling of a particular station. Additionally, **CHANGELIST** is used to restore polling of non-responding units.
- The **CLOSE** Procedure is used to terminate access to a line.

TRACE FACILITY

ENVOY provides an aid in checkout of communications applications with the Trace Facility. The Trace Facility works with all line types except auto call units.

Trace Facility records line "events" in a Trace Table. Each Trace Table entry provides;

- Sequence Number
- Controller and Line Number
- Line State
- Event
- Miscellaneous Information
- Timestamp

LINE STATISTICS

Line statistics are maintained by ENVOY for each line from the time a line is opened until the line is closed. The statistics are printed on the operator console when a predetermined error threshold is exceeded, when a path switch occurs, or when the line is closed.

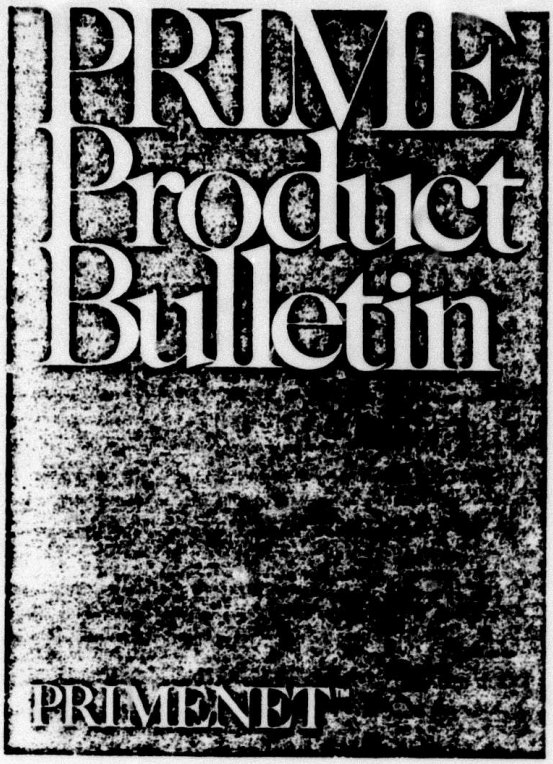
ON-LINE TESTING

ENVOYTST (Envoy Test) is used to verify the operation of the data communications process and the synchronous controller operating as the following line types:

- Point-to-point non-switched primary
- Point-to-point non-switched secondary
- Point-to-point switched primary
- Point-to-point switched secondary
- Multipoint supervisor
- Multipoint tributary

APPENDIX II

PRIME Network Systems



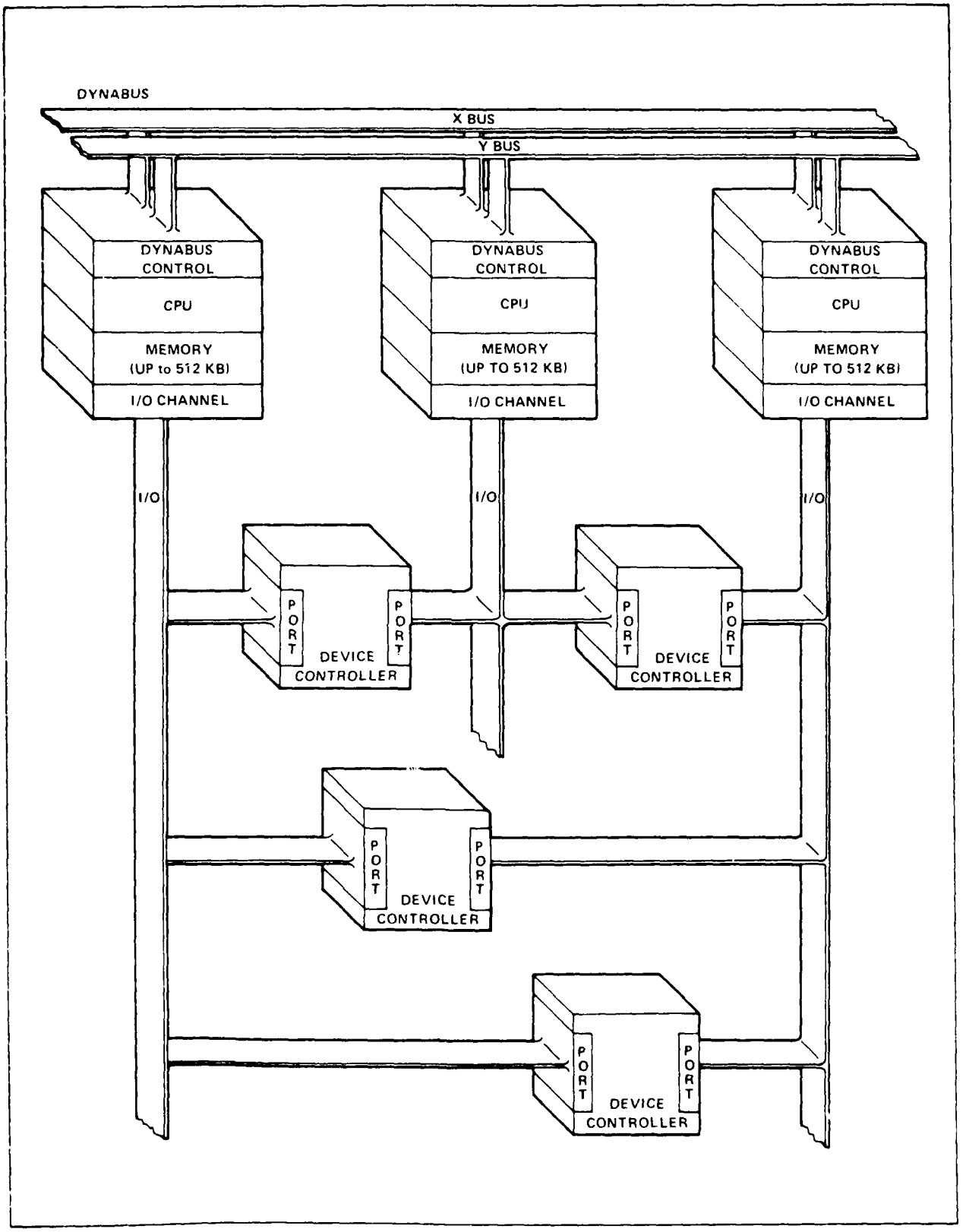
DESCRIPTION

PRIMENET™ provides complete local and remote network communication services for Prime systems. In geographically dispersed network configurations, it allows Prime computers to communicate with other Prime computers, with computers from other vendors, and with terminals attached to packet switching networks. In local network configurations, PRIMENET allows physically adjacent Prime computers to be attached via a high-bandwidth, multi-point loop arrangement.

PRIMENET software offers three distinct sets of services. The Inter-Program Communication Facility (IPCF) lets programs running under PRIMOS®, the Prime operating system, establish communications paths (virtual circuits) to programs in the same or another Prime system, or in other vendors' systems that support the CCITT X.25 standard for packet switching networks. The Interactive Terminal Support (ITS) facility permits terminals attached to a packet switching network, or to another Prime system, to log into a Prime system with the same capabilities they would have if they were directly attached to the system—but often at a significantly lower communication cost. And the File Access Manager (FAM), allows programs running under PRIMOS to utilize files physically stored on other Prime systems in a network. Remote file operations are logically transparent to the application program. Thus, users need not reprogram applications or learn new commands for network operation.

FEATURES

- Compatible with the Prime 750, 650, 550, 500, 450, 400, and 350 systems.
- Packet Network Interface option compatible with the CCITT X.25 packet switching interface standard, as presently implemented in the TELENET (United States) and DATAPAC (Canada) networks.
- Inter-Program Communication Facility.
- Interactive Terminal Support facility.
- Remote file access facility with transparent user interface.
- Network file security and access provisions.
- Primenet Node Controller (PNC), a direct, high-speed link between local systems, with sophisticated error recovery capabilities.
- High-speed, multi-line synchronous interface for remote communications, with hardware assists for X.25 Level II line protocols.
- Concurrent operation with remote job entry and Distributed Processing Terminal Executive (DPTX) software for mainframe communication.



Tandem 16 Computer System

The Tandem 16 Computer System fills three important and interrelated needs: it is a computer system where applications run *NonStop* regardless of a module failure, it provides a computer system that can support high transaction rates to large on-line data bases, and it provides a system that is easily adaptable to any application.

The basic design philosophy of the Tandem 16 Computer System is that no single module failure will stop or contaminate the system. This assurance of *NonStop* operation is sometimes called "fail-safe" when no loss of throughput occurs as a result of a failure or "fail-soft" when some slowdown occurs but full processing capabilities are maintained. The Tandem 16 can provide both "fail-safe" and "fail-soft' modes of operation.

- **PROCESSOR MODULES**

  A single Tandem 16 Computer System may contain from two to sixteen processor modules; each module contains a micro-programmed central processing unit, its own memory (up to 512k bytes), and its own micro-programmed input/output channel. Each processor module is fully capable of operating independently of all other processor modules yet can be configured to back up other processor modules.

- **INTERPROCESSOR BUSES (DYNABUS)**

  Each processor module is connected to all other processor modules via redundant high speed interprocessor buses. Programs running in one processor module communicate with programs running in other processor modules by means of these buses. Each interprocessor bus is fully autonomous, operating independently of (but simultaneously with) the other bus. The use of two buses assures that two paths exist between all processor modules in the system.

- **INPUT/OUTPUT CHANNEL**

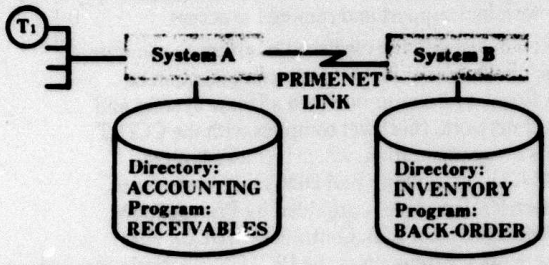
  Input/output devices (i.e, magnetic tape units, disc drives, terminals, etc.) are interfaced to the computer system via dual-port i/o controllers. Dual-port means that each i/o controller is connected to the input/output channels of two processor modules. This provides two paths of communication to each input/output device. A dual-port controller is "owned" by (i.e., will accept commands from) only one processor module. But in case of a failure, the other processor module can take control programmatically. The dual-port controllers are designed so that the number of components common to both paths are at a minimum.

- **TANDEM 16/TRANSACTION OPERATING SYSTEM (T/TOS)**

  Overseeing system operation is the Tandem 16 Operating System. The operating system provides the multiprocessing (parallel processing in separate processor modules), multiprogramming (interleaved processing in one processor module), and *NonStop* capabilities of the Tandem 16 Computer System. A copy of T/TOS resides in each processor module.

data elsewhere in the network. The same file protection mechanisms used in a single PRIMOS system apply to file operations performed over the network.

Because the interface to remote files is logically transparent to the user and to the program, the need to learn new command languages in order to use network resources is eliminated. Programs written to run in a single system environment automatically adapt to the network environment.



1. User at T1 accesses local file:		2. User at T1 accesses remote file:
ATTACH		ATTACH
ACCOUNTING	(local directory)	INVENTORY
OK,	(PRIMOS finds it)	OK,
COBOL	(invokes	COBOL
RECEIVABLES	COBOL compiler)	BACK-ORDER

Figure 4: Program-to-file communication. In case 1, the source file is resident on local system A; in case 2, the source file is resident on remote system B.

Figure 4 illustrates the dialogue used to compile a COBOL program when the source file is 1) resident on the local system, and 2) resident on another system in the network.

DISTRIBUTING PRIMENET FUNCTIONS

PRIMENET's architecture allows the system designer to distribute functions to individual systems in a network. Three major functions—terminal interfacing, file system operations, and application programs—can reside on the same system or be separated, as the following examples suggest.

Example 1: A computationally intensive program makes occasional references to remote files and terminals. Each function is assigned to a separate system: a Prime 550 handles terminal interfacing via ITS; a Prime 650 handles all file operations and is accessed via FAM; and a Prime 750 handles the intense computing load. The systems can be interconnected using PRIMENET Node Controllers or a packet switching network. Figure 5 illustrates this example.

Example 2: A program does a small amount of terminal I/O, and makes extensive use of the file system. Thus, it is advantageous to locate the application program and file system on the same computer, while allowing terminals on other systems to access the

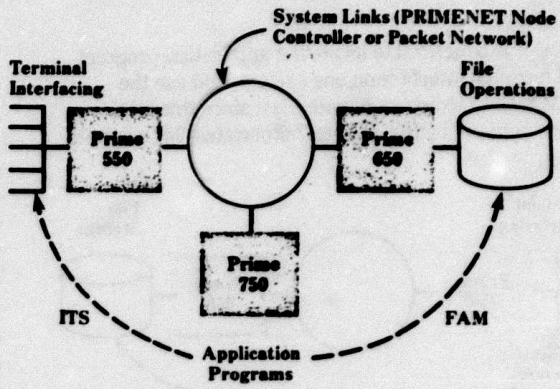


Figure 5: In this example network, the Prime 550 handles terminal interfacing via ITS; the Prime 650 handles file operations and is accessed via FAM; and the Prime 750 handles the intense computing load. The dotted lines represent PRIMENET functions.

application via ITS, since this minimizes the amount of network data traffic. For instance, it is preferable to compile a source program on the same system where the source program is stored, instead of reading the entire source program into a different system, compiling it, and writing the object program back to the second system. Access to the compiler from other systems is provided by ITS.

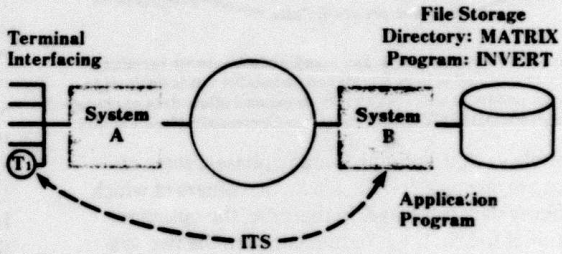


Figure 6: When a program does a small amount of terminal I/O, and makes extensive use of the file system, it is best to locate the program and file system on the same computer, and to allow terminals on other computers to access the application via ITS.

Referring to Figure 6, the following dialogue is used on terminal T1 to compile the FORTRAN source program, INVERT, stored on System B, using the FORTRAN compiler also on System B:

```
LOGIN account -ON SYSB....use ITS to get to
                          System B
OK,
ATTACH MATRIX.....locate directory
                  MATRIX
OK,
FTN INVERT.....compile program
```

Example 3: A program has many interactive exchanges with terminal users, generates high-volume terminal output, but requires little access to the file system.

Thus, it is beneficial to locate the application program and terminal interface on one system, and use the FAM facility to access remote files, since this minimizes network traffic. Figure 7 illustrates this example.

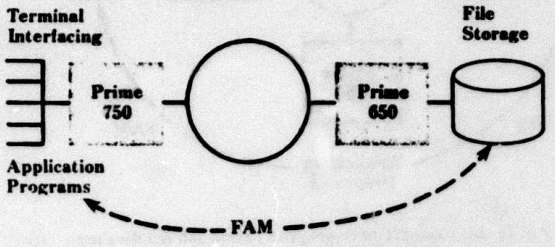


Figure 7: When a program requires little access to the file system, yet has many exchanges with terminal users, it is advantageous to use FAM to access remote files because it minimizes network traffic.

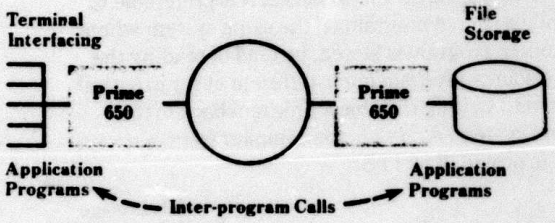


Figure 8: When a program has many phases—some terminal I/O intensive and others file system intensive—it is optimal to split the program between two systems and allow data exchange through PRIMENET's Inter-program Communications Facility.

Example 4: A program has many phases, some of which are terminal I/O intensive, and others of which are file system intensive. In this case, the optimum solution is to split the program between the two systems, with the component in one system handling complex terminal interactions, and the component in the other system handling complex file system interactions. PRIMENET's Inter-Program Communication Facility is used to exchange data between the two component programs, as shown in Figure 8.

PRIMENET ARCHITECTURE

PRIMENET adapts easily to new standards and technologies because it uses a layered architectural design. The software that supports each element of PRIMENET is separate from each other element, and communicates with the others via well-defined interfaces. This means PRIMENET can support a variety of individual communications protocols, increasing its flexibility in different configurations. Protocol layering also makes it possible for Prime to continuously evolve PRIMENET products to take advantage of new technologies and emerging standards—without disturbing the user environment for existing PRIMENET applications.

PRIMENET contains four levels of facilities. The first three correspond to the CCITT X.25 recommendation, and relate to the network's electrical interface, link level protocols, and packet level procedures. The fourth level provides user services, including interactive terminal support and remote file access.

LEVEL I deals with the electrical interface to communications media. For synchronous communication between Prime systems or between a Prime system and a packet network, this level complies with the CCITT X.21bis recommendation, which includes both the RS-232-C (V.24) and the Bell DDS (V.35) interfaces. The electrical interface is provided by Prime's high-speed Multiple Data Link Controller (MDLC). For local networks, Prime offers the PRIMENET Node Controller (PNC), which includes a unique electrical interface to a high-speed coaxial cable.

LEVEL II provides management services for the communication link attached to the PRIMENET package. With the Packet Network Interface, this level complies with CCITT X.25 LEVEL II procedures, which use the HDLC protocol in full-duplex, Asynchronous Response Mode (ARM). A full-duplex binary synchronous framing procedure is available in North America for connection to the TELENET and DATAPAC public networks, or for connecting two Prime systems via a full-duplex synchronous facility. When attached locally via the Primenet Node Controller, Prime systems use a unique hardware-implemented protocol, which provides data integrity via cyclic redundancy checking.

LEVEL III, when augmented with the Packet Network Interface, corresponds to CCITT X.25 LEVEL III, and deals with the management of virtual circuits between a Prime system and others in network. LEVEL III services include virtual circuit establishment and clearing, multiplexing/demultiplexing of many virtual circuits onto a single physical facility, packet sequencing through the network, flow control procedures, and network addressing. This layer is accessible to PRIMOS user processes via a set of inter-program calls. Both FAM and ITS use LEVEL III facilities as their data transport mechanism.

PRIMENET LEVEL IV software provides network user services, including file access and terminal support. The File Access Manager (FAM) resides above LEVEL III, and connects the PRIMOS file management system to the network. FAM accepts requests both from the local file system for access to files located elsewhere in the network, and from remote file systems for access to local files.

LEVEL IV also includes the Interactive Terminal Support (ITS) facility, which provides data paths between the PRIMOS terminal manager and the network.

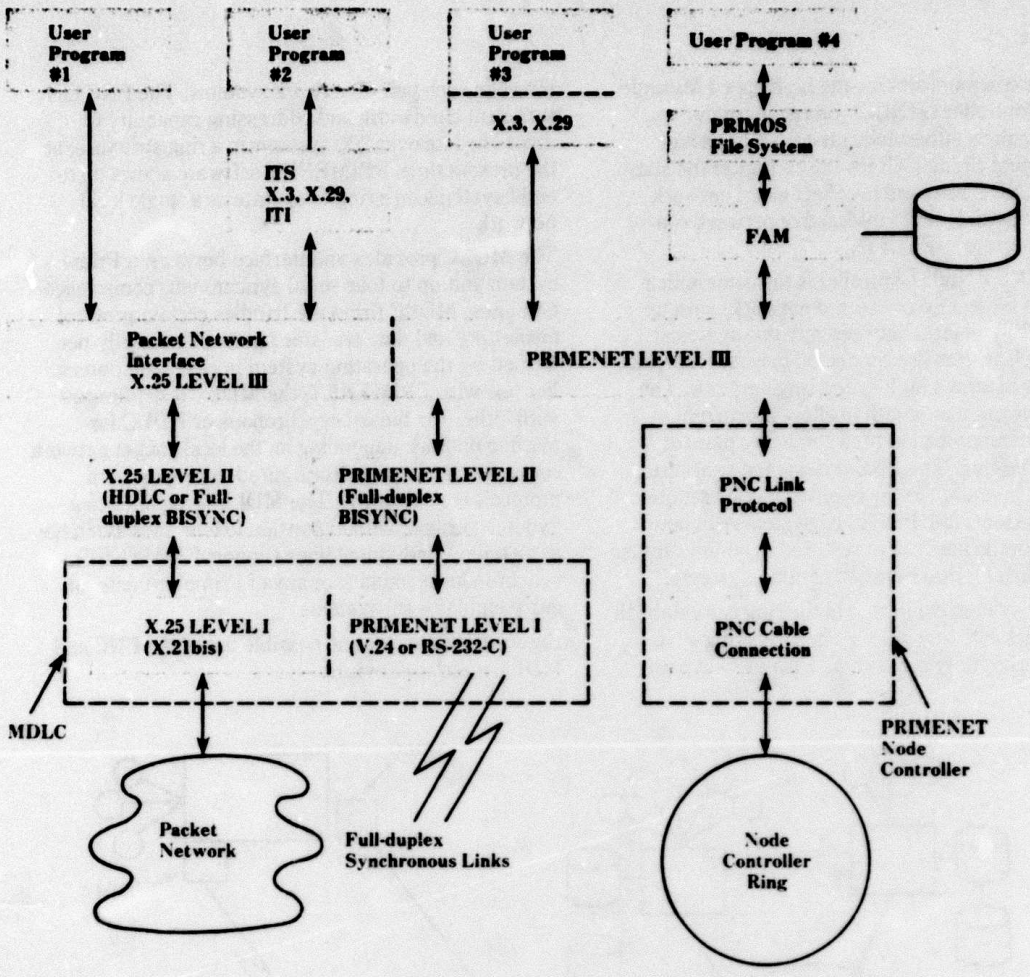


Figure 9: PRIMENET's layered architecture. The Packet Network Interface augments PRIMENET for operation over packet switching networks; all PRIMENET functions (IPCF, ITS, and FAM) can be used over all communications media. For example, user program #1 is communicating with a program in another system via IPCF. User program #2 is interacting with a user at a terminal on another system or on the packet network. User program #3 includes its own copy of the terminal control protocols, and can communicate simultaneously with many remote terminal users. User program #4 utilizes both local and remote files in a transparent manner. A single user program con combine these functions in any required manner.

ITS complies with CCITT X.3 and X.29 procedures, which describe the operation of asynchronous terminals in the packet network environment. In North America, PRIMENET also supports the Interactive Terminal Interface (ITI) protocols defined by TELENET and DATAPAC. ITS allows terminals physically attached to a packet network to interact with a Prime system, and vice versa.

Figure 9 illustrates the layered architectural arrangement of PRIMENET software.

FILE SECURITY AND ACCESS CONTROL

PRIMENET furnishes the same level of security and access control for files accessed through the network as it does for files access locally. Each User File Directory (UFD) in a network, regardless of its location, contains both owner and non-owner specifications for operations such as read/execute, write, and truncate/delete. In addition, a system administrator can define certain regions of disk storage to be available only to local users, preventing access to files in these regions by remote programs.

FLEXIBLE NETWORK CONFIGURATION

PRIMENET users can choose from a range of systems including the Prime 750, 650, 550, 500, 450, 400, and 350. For local networks, Prime offers the high-performance PRIMENET Node Controller, and for

synchronous communications, the high-speed Multiple Data Link Controller (MDLC). Common carrier connections can be either dedicated lines or packet switched virtual circuits. Users can configure the standard PRIMENET software to reflect exact network requirements with Prime's interactive network configuration program, NETCFG.

The PRIMENET Node Controller is the communication medium for locally connected networks with up to 750 feet (230 meters) between any two systems. PNCs for each system are connected to each other with coaxial cable to form a high-speed ring network. The ring has sufficient bandwidth to allow concurrent high-speed communication between many pairs of systems on the ring. The PNC is designed to assure continuity of operation in the event that one or more individual systems fail. Individual systems can be removed from the network or restored to on-line status, without disturbing the operation of other systems.

Further, any system connected in the ring can establish virtual circuits with any other system, making PNC-based networks "fully connected" with a direct path between each pair of network systems. The PNC has sufficient bandwidth and addressing capability to accommodate over 200 systems in a ring structure; at the present time, PRIMENET software allows up to eight systems on a ring to operate as a single local network.

The MDLC provides an interface between a Prime system and up to four serial synchronous communication lines. MDLC firmware handles certain protocol formatting and data transfer functions normally performed by the operating system in other computers. For use with PRIMENET, the MDLC is configured with either the binary synchronous or HDLC formatting options, depending on the local packet network conventions. (It can be configured to support both protocols concurrently.) The MDLC is supplied in a two-line version, with expansion to four lines available as an option. Individual lines support PRIMENET, synchronous terminals, or any of Prime's remote job entry emulator subsystems.

Figure 10 illustrates one possible use of the PNC and MDLC interface devices.

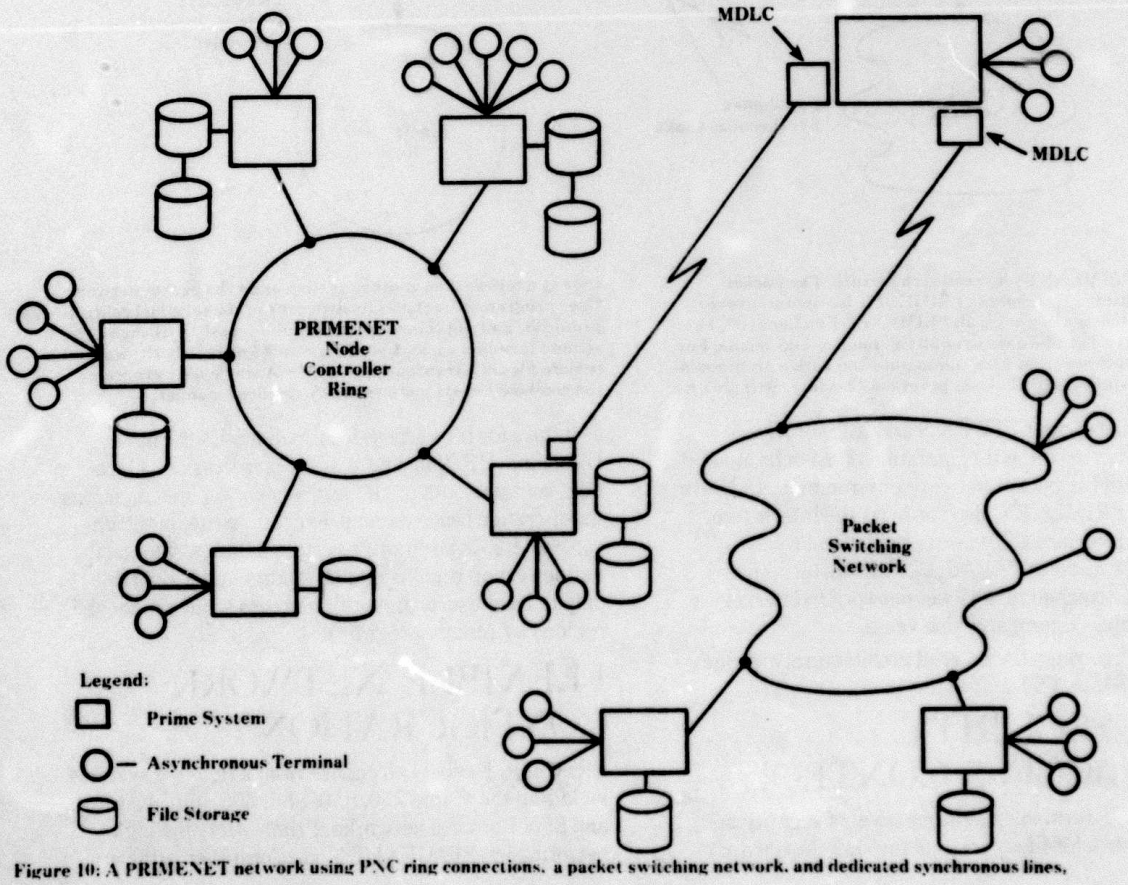


Figure 10: A PRIMENET network using PNC ring connections, a packet switching network, and dedicated synchronous lines.

MAINFRAME-COMPATIBLE SOFTWARE

Prime computers can communicate with mainframes provided by other vendors, concurrently with PRIMENET operation. Prime's remote job entry emulation packages support IBM 2780 (Model 1) and HASP, Control Data 200UT, Univac 1004, and ICL 7020. Prime also offers the Distributed Processing Terminal Executive (DPTX), which conforms to protocols used by IBM 3271/3277 Display Systems. These packages let mainframe users offload many processing functions, typically of an interactive nature, to smaller, more cost-effective Prime systems.

FURTHER INFORMATION

PRIMENET is a licensed software product that includes all PRIMENET functions (IPCF, ITS, and FAM) for operation with the PRIMENET Node Controller and full-duplex synchronous facilities. The Packet Network Interface is an additional licensed software product that augments the operation of PRIMENET for communications through packet networks supporting the CCITT X.25 standard. PRIMENET and the Packet Network Interface operate under PRIMOS on the Prime 750, 650, 550, 500, 450, 400, and 350 systems. For further information, consult your local Prime Marketing Representative.

PRIMENET

Printed in U.S.A. Specifications subject to change without notice. NS40056 98.

PRIME

PRIME Computer, Inc., 40 Walnut Street, Wellesley Hills, MA 02181

Sales and service offices in major U.S. cities with subsidiaries in Denmark, Federal Republic of Germany, France, Norway, Sweden, and the United Kingdom; distributors in Australia, Austria, Canada, Ecuador, Greece, Italy, Japan, Korea, The Netherlands, New Zealand, Saudi Arabia, Singapore, and Switzerland.

PRIME Product Bulletin

Communication Subsystems

DESCRIPTION

Prime's communication subsystems offer high performance and reliability when using Prime systems for communications applications. The subsystem series includes three hardware devices that operate as interfaces for Prime systems and peripherals in local and/or remote locations.

The Asynchronous Multi-Line Controller (AMLC) allows Prime systems to control multiple asynchronous communication lines. Keys to its high performance are microprogrammed architecture and Direct Memory Queue (DMQ) operation, which reduce overhead and increase system throughput.

The Multiple Data Link Controller (MDLC) is a hardware interface for synchronous applications. It provides support for a variety of protocols for flexibility in implementing communication networks. Among them are IBM BISYNC, HDLC for X.25 packet switching networks, CDC 200UT, and Univac 1004.

The PRIMENET™ Node Controller (PNC) is a communication medium for locally connected networks. Through a sophisticated distributed control design, it assures reliable network operation.

ASYNCHRONOUS MULTI-LINE CONTROLLER (AMLC)

The high-performance Asynchronous Multi-Line Controller (AMLC) is a hardware interface that lets Prime systems control multiple asynchronous communication lines. It increases system throughput and reduces overhead with microprogrammed architecture and the Direct Memory Queue (DMQ) mode of operation. The DMQ technique takes advantages of central processor microcode to provide fast, efficient buffering of output data. This mode of operation allows the AMLC to queue messages without the need for extensive software management, and operates autonomously to process characters, thus reducing the number of interrupts that must be serviced by the central processor.

The AMLC provides full-duplex communication support, plus a number of programmable line characteristics that simplify configuring networks with mixed equipment. Under program control, each AMLC line can select one of eight clock speeds from 110 to 9,600 bps. AMLC lines connect to RS-232-C/CCITT V.24 or 20 mA current loop-compatible terminals and peripheral devices, as well as Bell 103, 113, and 212 data sets.

AMLC Features

- Fully supported by PRIMOS*.
- Microprogrammed architecture and Direct Memory Queue (DMQ) mode of operation reduce overhead.
- Program-selectable line speed allows easy implementation of communication systems without hardware changes.
- Easy configuration of networks with mixed equipment through support of full-duplex operation; EIA RS-232-C/CCITT V.24, and 20 mA current loop interface standards; and line speeds to 9,600 bps.
- Available in eight-line or 16-line versions.

MULTIPLE DATA LINK CONTROLLER (MDLC)

The high-speed Multiple Data Link Controller (MDLC) is a hardware interface for synchronous communication applications using Prime systems. The MDLC lets users interface Prime systems to communication lines with support for multiple protocols: IBM BISYNC for HASP and 2780; High-Level Data Link Control (HDLC) protocol for CCITT X.25 packet switching networks; Control Data 200UT; Univac 1004; and ICL 7020.

By using microcode to format communication messages and to test data for special characters and sequences, the MDLC increases system throughput. Direct Memory Access (DMA) and full character buffering are two additional facilities that provide high-speed operation and low overhead.

The MDLC features Cyclic Redundancy Checking (CRC) and parity checking for error detection and correction, plus a line-looping feature for hardware troubleshooting. For flexibility in communication system development and adaptation to new equipment, a number of line characteristics are user-programmable.

MDLC Features

- Fully supported by PRIMOS.
- Microprogrammed architecture increases throughput

AMLC and MDLC: Summary of Features

	AMLC	MDLC
Transmission type	Asynchronous	Synchronous
Lines per board	8 or 16	2 or 4
Line speeds	110 to 9,600 bps	2,000 to 19,200 bps
Line interfaces	EIA RS-232-C/CCITT V.24 or 20 mA	EIA RS-232-C/CCITT V.24
Program selectable character size	8 bits with 1 or 2 stop bits	5, 6, 7, or 8 bits
Model control (Bell or equivalent)	103, 113, 212	201, 203, 208, 209, or DDS
Model control signals	Carrier Detect, Ring Indicator, Data Set Ready, Request to Send, Data Terminal Ready, Clear to Send	Carrier Detect, Data Set Ready, Request to Send, Data Terminal Ready, Clear to Send, Speed Select (for loop-back operations)
Program-selectable parity	Odd, even, or none	Odd, even, or none
Error detection scheme	Parity	CRC-12, -16, -24, CRC-CCITT, or LRC-8
Protocols supported	Not applicable	IBM BISYNC, HDLC X.25, CDC 200UT, Univac 1004, ICL 7020

by eliminating many tasks from central processor overhead.
- Direct Memory Access (DMA) reduces software intervention by executing data transfers independently of the central processor.
- Support for multiple protocols enhances versatility in communication environments with mixed equipment.
- Support of EIA RS-232-C and CCITT V.24 electrical interface standards at speeds to 19,200 bps, in half- of full-duplex mode.
- Programmable line characteristics assist in incorporating new network devices.
- Cyclic Redundancy Checking (CRC) and parity checking ensure communication link integrity.
- Available in two-line or four-line versions.
- Local and remote loop-back features simplify hardware fault diagnosis.

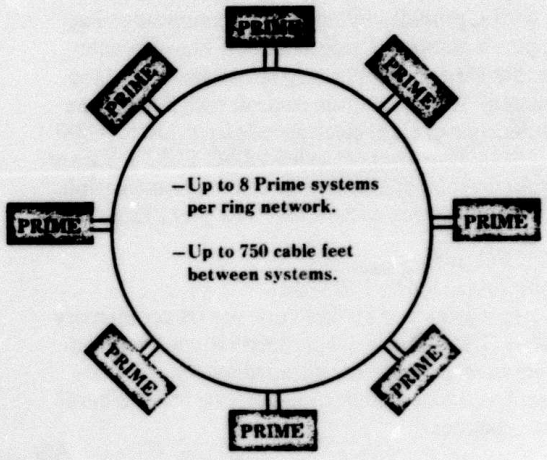
PRIMENET NODE CONTROLLER (PNC)

The PRIMENET Node Controller (PNC) is a communication facility for locally connected networks. To assure continuity of operation in the event that one or more individual systems fail, it uses a sophisticated distribution control algorithm. Thus, individual systems can be removed from the network or restored to on-line status, without disturbing the operation of other systems, providing the system-to-system distance criteria are met. PNCs for each system are connnected with coaxial cable to form a high-speed ring network, with up to 750 feet (230 meters) between any two systems.

Any system in the PNC ring can establish virtual circuits with any other system, making PNC-based networks "fully connected" with a direct path between each pair of systems. The PNC has sufficient bandwidth and addressing capability to accommodate over 200 systems in a ring structure; however, at the present time, PRIMENET software supports up to eight systems on a ring to operate as a single local network.

Messages on the PNC network are sent as one or more packets that consist of header, variable length data field, and a Cyclic Redundancy Checked (CRC) trailer. The packet can have a data field of up to 2044 bytes, plus the four byte header. CRC ensures data integrity so that the data sent to the receiving processor is identical to the data sent from the transmitting processor. An active system in the network is unaware of messages destined for other systems, and the central processor is notified only when a message for that system has been correctly received.

PNC Features
- Fully supported by PRIMENET, Prime's networking software.
- Allows fully connected networks of up to eight Prime systems (nodes) with only a single controller on each system.
- Up to 750 feet (230 meters) between active nodes.
- High-performance packet transmission at speeds up to eight million bits per second.
- Failsoft operations provided through distributed control architecture. Hardware or software failures, including loss of power, in one node do not affect network operations between other active nodes.
- Link protocol, implemented in hardware, ensures data integrity via CRC mechanism, with no central processor overhead.
- An active node is unaware of messages destined for other nodes in the network, and the central processor is notified only when a message for that node has been correctly received.
- Loop-back facility for diagnostic purposes.

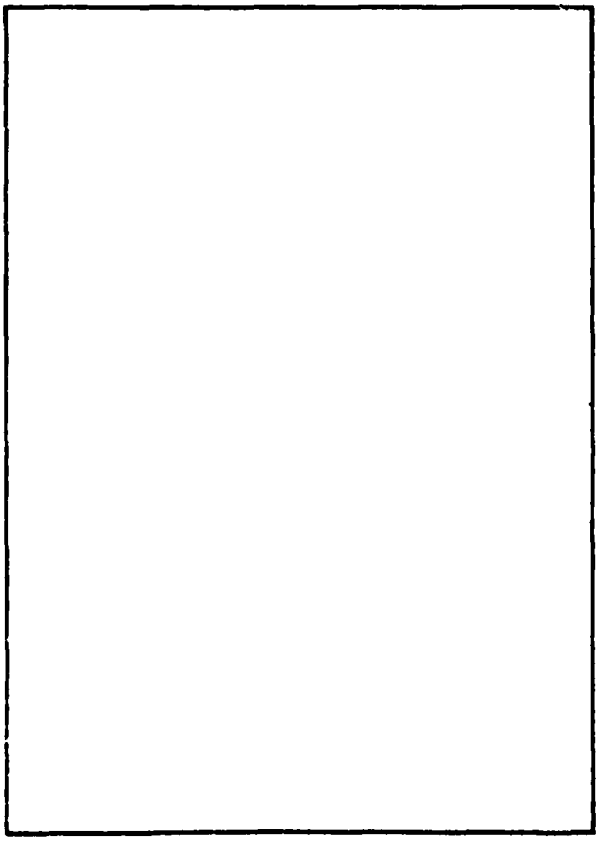


FURTHER INFORMATION

The AMLC, MDLC, and PNC are hardware products for the Prime, 750, 650, 550, 500, 450, 400, and 350 systems. For futher technical or ordering information, consult your local Prime Marketing Representative.

PRIME Product Bulletin

Prime 750 System



DESCRIPTION

The Prime 750 is the most powerful member of Prime's new family of hardware- and software-compatible systems. With many features that fully exploit its 32-bit architecture, the 750 offers speed, low overhead, and flexibility in a wide range of computational timesharing and interactive data processing applications.

Key contributors to the Prime 750's speed and low overhead are a 16K-byte cache memory, instruction prefetch unit, high-bandwidth burst mode I/O, and interleaved main memory. In addition, a high-performance floating-point unit provides instruction execution speeds that rival systems costing much more. This feature, along with a 32 million-byte virtual address space for programs, makes the 750 a fast and powerful tool in scientific and engineering applications.

For commercial applications, the Prime 750 offers COBOL program execution speed exceeded by no system short of a mainframe. Especially in environments requiring the flexibility to use other languages, and add users and applications, the 750 outperforms any system in its price range.

Programs written for any Prime system run without modification on the 750. And since it is compatible with all Prime peripherals, controllers, and I/O interfaces, any Prime system can be upgraded to a 750 at a fraction of the total system cost and with minimal conversion effort.

FEATURES

- Embedded operating system, PRIMOS®, for fast access to all system resources, timesharing for up to 63 users, batch, and multi-tasking.
- 32 million-byte virtual address space per user.
- 16K-byte, 80 nanosecond cache memory.
- Instruction prefetch and decoding.
- High-bandwidth burst mode I/O.
- 32-bit memory with 64-bit interleaved memory data transfers.
- Error-correcting MOS main memory expandable to 8 million bytes.
- High-performance floating-point arithmetic unit.
- Hardware-implemented business instruction set.
- High-speed, 32-bit integer arithmetic unit.
- 128, 32-bit hardware registers for system management.
- Extensive parity checking throughout the processor and for each microcode control word, and enhanced error reporting.
- Hardware-implemented rings of protection for system and user software security.
- Virtual control panel for remote hardware and software troubleshooting.

PRIME 750: MINICOMPUTER RESPONSIVENESS AND MAINFRAME CAPABILITY

The Prime 750 integrates the proven attributes of the previous top-of-the-line system, the Prime 500, with many new features that optimize its 32-bit internal architecture. The result is an exceptionally fast, responsive computer system with mainframe functionality. Yet, the Prime 750 is both affordable and totally compatible with its predecessors.

At the heart of the 750's speed are a high-capacity cache memory, instruction prefetch buffer, burst mode I/O, interleaved main memory, and a high-performance floating-point unit. Its efficient microcode structure ensures high-speed instruction execution. The number of microcode steps, as well as the time required for each step in an instruction execution cycle, are significantly lower than other systems. The 750, for example, completes a 32-bit register memory add instruction in but one machine cycle (micro-step).

The Prime's 750's mainframe-level capabilities are a derivative of PRIMOS, the uniform operating system for all Prime computers. With PRIMOS, the 750 can perform timesharing for up to 63 users, multi-tasking, and batch operations. Each user has a 32 million byte virtual address space and a variety of languages that simplify programming.

HIGH PERFORMANCE, LOW OVERHEAD

Four facilities contribute to the Prime 750's speed and efficiency: a high-capacity cache memory, an instruction prefetch unit, burst mode I/O, and interleaved main memory.

Cache Memory

With a 16K-byte capacity, the Prime 750 cache memory dramatically reduces instruction execution times and processor overhead. This high-speed (80 nanosecond access) bipolar memory decreases the effective memory cycle time to near that of the processor by storing the information that most likely will be used next by the processor. Its high capacity provides a 95 per cent hit rate on anticipated processor requirements.

Both the instruction prefetch unit and the central processor's execution unit access cache memory. Cache is located on the central processor, rather than on main memory boards, thereby eliminating memory bus delays in cache-to-processor transfers. Memory mapping and cache memory usage completely overlap, further reducing execution time and central processor administrative overhead.

Instruction Prefetch Unit

The instruction prefetch unit improves central processor performance by both prefetching and decoding up to four instructions from cache memory. Because it accesses cache independently of the central processor, it prefetches, decodes, and forms the effective address in parallel with instruction execution. Therefore, when the processor's execution unit is ready for the next sequential instruction, the instruction is prepared for execution in advance. This means the Prime 750 spends more time executing and less in time-consuming administrative overhead.

Control logic in the instruction prefetch unit continuously fetches data from cache to keep its buffer full. In addition, the instruction prefetch unit can resolve most indirect addresses, further increasing instruction execution speed.

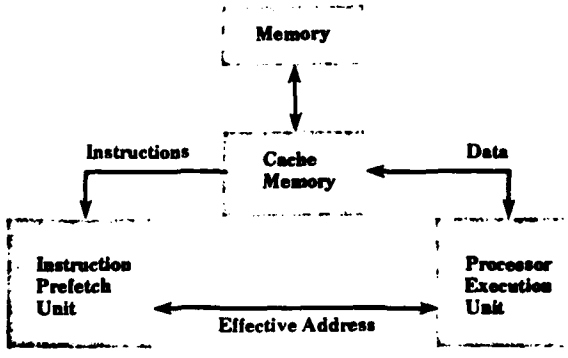


Figure 1: The instruction prefetch unit accesses cache memory independently of the central processor to perform instruction prefetch and decoding in parallel with execution.

Burst Mode I/O

The Prime 750 uses an I/O mode that increases throughput and reduces central processor overhead. Called burst mode I/O, it provides an 8 million-byte-per-second transfer rate over the interface between the processor and peripheral controllers. This high bandwidth channel reduces the central processor's I/O load, and enhances the performance of high-speed peripherals.

The Prime 750 supports direct-to-memory I/O operations with four types of access modes: Direct Memory Access (DMA), Direct Memory Control (DMC), Direct Memory Transfer (DMT), and Direct Memory Queue (DMQ).

The 750 has 32 DMA channels, controlled by high-speed channel address registers, which support an 8 million-byte-per-second transfer rate using burst mode I/O. High-speed peripherals such as disk subsystems typically use these channels.

DMC channels, controlled by channel address words in the first 8K-bytes of virtual memory, offer up to 2048 channels for medium-speed I/O transfers such as those

in serial data communications. Their maximum transfer rate is 960K bytes per second.

DMT handles channel programs for high-speed device controllers, such as the controllers for moving-head disks. Its maximum throughput rate is 2.5 million bytes per second.

In addition, DMQ channels provide circular queues for handling communication devices. These queues reduce operating system overhead by eliminating interrupt handling on a character-by-character basis.

Interleaved Main Memory

The Prime 750, like all other Prime systems, uses MOS main memory exclusively. It can address up to 8 million bytes of ECC main memory, and reduces main memory access time to near that of the central processor through the use of the 16K-byte cache memory.

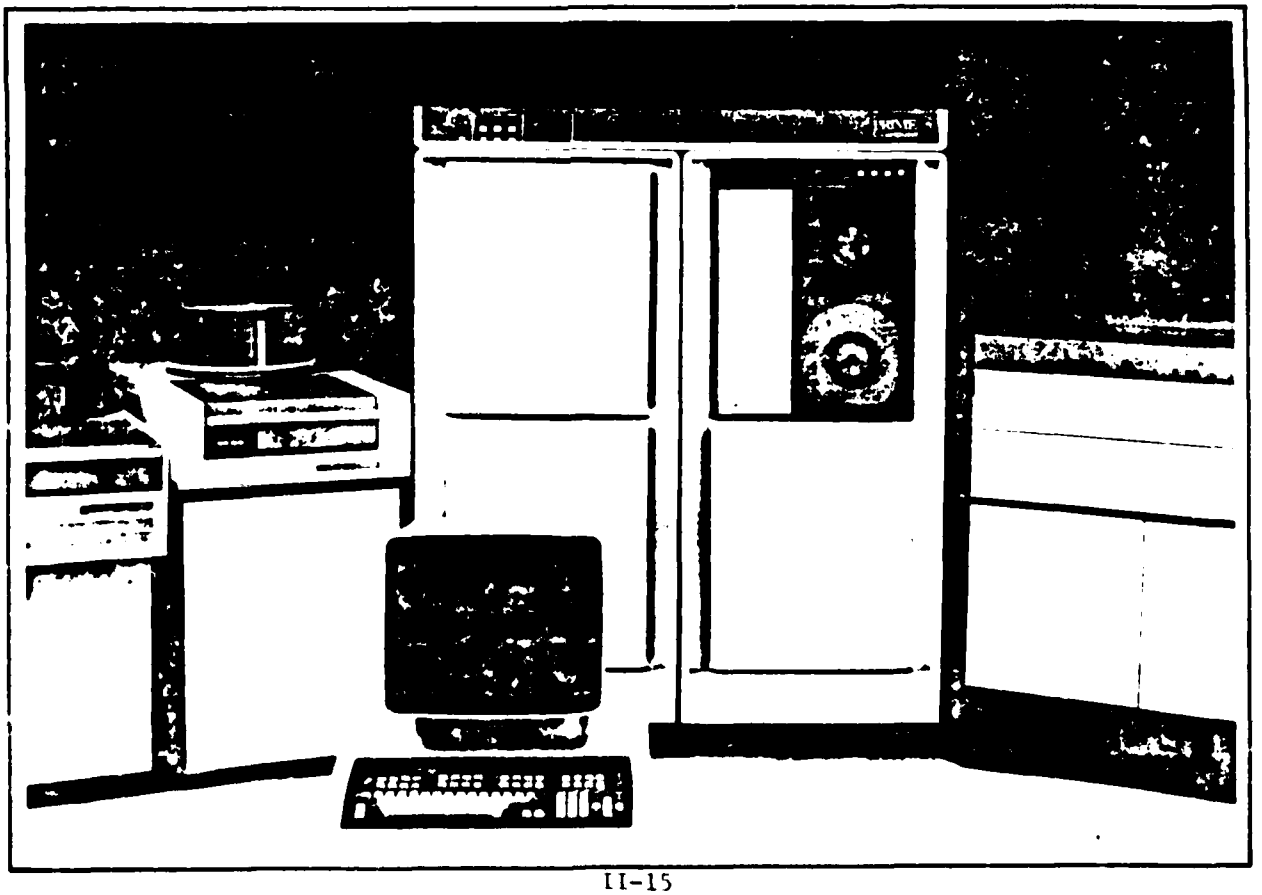
Because consecutive memory locations are on separate memory boards, interleaving can be used to speed-up sequential memory accesses and maximize the cache hit rate. In effect, interleaving provides high-speed transfers between memory and the central processor by allowing the processor to read or write four bytes at a time. During burst mode I/O, interleaving provides 64-bit data transfers for optimum performance of high-speed disks and magnetic tape subsystems.

In addition to 32-bit transfers and 64-bit interleaving, the 750 supports 16-bit transfers for I/O devices and controllers that use this format, as well as half-word instructions such as a 16-bit WRITE. This preserves existing investments in peripheral equipment and maintains total compatibility of programs.

VIRTUAL MEMORY MANAGEMENT

The Prime 750's virtual memory management facilities simplify program sharing and give users a powerful 32 million-byte virtual address space for programs. This lets programmers write large programs that can execute in both small and large main memory configurations with no need to define overlays or later modify programs to take advantage of additional memory.

Memory management facilities provide each system user with a private and shared system address space. Through segmentation and paging, this space is divided into 128K-byte segments, of which 256 segments (32 million bytes) are available for each user, and the rest for shared operating system software. By embedding operating system functions in each virtual memory space, all operating system functions are immediately available as if they were an integral part of a user's program, dramatically reducing system overhead.



SYSTEM ARCHITECTURE

Prime was the first company to put software first in the natural order of engineering. The Prime 750 is an extension of that philosophy because its hardware is designed to complement system software. For example, the 750's program environment, control and scheduling facilities, and resource allocation mechanism operate within the framework of the PRIMOS operating system. The result is a system that decreases the cost of computing and increases application flexibility.

Uniform System Software
A single, uniform operating system, PRIMOS, lets all Prime systems perform multi-terminal, batch, and multi-task operations. Because PRIMOS is embedded in each user's virtual memory space, it ensures fast program access to operating system resources. It supports re-entrant procedures, permitting a single copy of a software module, such as the text editor or FORTRAN compiler, to be shared by many users. PRIMOS further supports ANSI '74 COBOL, ANSI '77 FORTRAN, BASIC, BASIC/VM, RPG II, PL/1, Prime Macro Assembler, the source-level debugger, and the various levels of query facilities provided by PRIME/POWER; DBMS, Prime's CODASYL-compatible Database Management System; MIDAS, the Multiple Index Data Access System; FORMS, the Forms Management System; and a wide variety of application packages available from the Prime Users Library Service (PULSE), from other users, and from third party vendors.

Stack Architecture
Prime 750 programs operate in a multi-segment environment including a stack segment that contains all local variables; an instruction or procedure segment, and a linkage segment that contains statically allocated variables and linkages to common data. Highly efficient addressing modes provide access to stack and linkage variables. Hardware-implemented CALL and RETURN instructions eliminate the overhead of software stack management routines. The 750's stack architecture optimizes the efficiency of operations such as parameter passing, subroutine and procedure calls, arithmetic expression evaluation, and dynamic allocation of temporary storage.

Process Exchange
A process exchange facility automatically transfers the central processor's attention from one activity (process) to another, with minimum overhead and complete protection. It allocates resources to the highest priority process in a series of tasks awaiting execution in a queue. It further handles the logistics of processes ready for execution and those waiting for a specific event to occur. The process exchange facility includes firmware that automatically dispatches a task for execution and reorders those remaining without software intervention.

Register Sets
The Prime 750 has 128, 32-bit hardware registers divided into four sections. One 32-register section handles firmware and operating system functions. The second section controls the processor's 32 high-speed Direct Memory Access (DMA) channels. The remaining two sections hold the machine states of active processes. The process exchange facility manages register assignment to processes dynamically and automatically.

SYSTEM INTEGRITY

The Prime 750 provides system integrity through comprehensive error detection and reporting mechanisms. Microverification routines are invoked automatically when the system is initialized. These routines test the validity of the CPU logic and indicate the cause of any malfunction via a diagnostic status word.

While the system is running, parity checking ensures data integrity throughout the processor's internal busses, registers, and other data paths. In addition, the 750 automatically checks the parity of each microcode control word. Error-correcting code in real memory automatically detects and corrects all single-bit errors, making them totally transparent to the users. All two-bit errors are reported as well.

The Prime 750 also includes a comprehensive, hardware-controlled memory protection system. A multi-ring protection hierarchy allows programs to be assigned to any of several security levels. Thus, multiple users can have full access to specified programs, while other programs and databases are protected against unauthorized access, and operating system software is guarded against accidental user intrusion.

FAST, FLEXIBLE COMPUTATION

The Prime 750 gives users the flexibility to perform a variety of arithmetic operations on three generic data types: floating-point, decimal, and integer. Its new floating-point unit provides execution speeds comparable to much more expensive systems. With the expanded business instruction set, the 750 offers very fast decimal arithmetic. It provides fast integer arithmetic with a 32-bit unit built to support system architecture. And its basic instruction set includes many capabilities that are typically a part of the operating system or are subroutines in other computers—for example, queue handling, parameter passing, and conditional branching.

These microcode-implemented facilities are designed to optimize PRIMOS, the compilers supported by PRIMOS, and the system's 32-bit architecture. The result is speed and simplicity in accessing system resources, manipulating data, and program control.

Floating-point Arithmetic
With the Prime 750, users can expect floating-point performance that exceeds much more expensive systems. Its double-precision throughput ranks among the highest in the industry. And in timeshared, multilingual environments, the Prime 750 far surpasses the floating-point performance of any system in its price range.

A new floating-point unit that optimizes the 750's 32-bit architecture is the major contributor to this performance. With a 32-bit path between the unit and central processor, the unit can accept data at an extremely fast rate. Registers assigned to floating-point operations are integral to the unit itself. Its use of parallel logic permits exponential and fractional calculation to be done simultaneously, still another reason for its speed. Separate parallel logic performs binary multiplication four bits at a time, division three bits at a time, and addition 48 bits at a time.

Business Instructions
The Prime 750 provides high-level support for ANSI '74 COBOL with comprehensive instructions designed for decimal arithmetic, character field manipulation, and editing operations. Compared with even the impressive COBOL performance of the previous top-of-the-line Prime 500, it increases COBOL throughput by a factor of two.

COBOL decimal arithmetic operations support packed or unpacked signed numbers of up to 18 digits. Operands differing in data type and/or scale factor are handled automatically during add, subtract, multiply, divide, and comparison operations. Business instructions also allow rounding on numeric operations, and provide for binary-to-decimal and decimal-to-binary conversions.

Character operations can be applied to field sizes of virtually any length. Justification, truncation, and padding are automatic in move, compare, translate, edit, and similar operations. Numeric and character editing instructions let the user easily produce fields in ANSI '74 COBOL picture-like formats. For maximum efficiency, all business operations involve a limited number of in-line instructions.

High-speed Integer Arithmetic Unit
All integer arithmetic and logical operations are performed in the processor's 32-bit arithmetic unit. By using a 32-bit format, the Prime 750 significantly improves the execution times of integer arithmetic instructions. The design of the arithmetic unit also permits efficient handling of complex formation, such as base-plus-displacement and indexing.

Standard Instruction Set
The standard instruction repertoire is a compatible superset of the machine instructions available with previous Prime systems. Its addressing mode compatibility allows user programs written for any multiuser Prime system to run without modification on the 750.

Over 300 instructions provide enhanced operating system communication, data handling, and cooperation of processes. Highly flexible address formation techniques allow all instructions to use any of four user-accessible base registers, up to seven index registers, and 32-bit indirect words in any combination. Thus, all memory reference instructions can access any datum in the entire virtual address space.

The Prime 750 features instruction capabilities that exploit the system's 32-bit memory and cache data paths, and 32-bit internal architecture. Of its eight 32-bit general purpose registers, seven can be used as index registers. These registers also can be used in local storage for compiler optimization and as fixed-point and logical accumulators. There are two floating-point registers, each 64 bits long, and two field address and length registers used by character and decimal instructions, also 64 bits long. Four other 32-bit registers include a procedure base, stack base, link base, and auxiliary base register.

MAXIMUM UPTIME WITH EASY, REMOTE DIAGNOSTICS

The Prime 750 includes a sophisticated Virtual Control Panel (VCP). It allows a diagnostic specialist to remotely control any system. This provides users with fast, effective troubleshooting—whether the task is identifying a hardware problem or installing a new revision of the operating system.

The system administrator permits remote access by simply pushing a button. A second button provides one of two modes of remote access: the remote terminal can be placed in monitor mode only, or be given the capability to operate as if on-site.

The VCP displays the state of the remote communications link via two indicator lamps mounted on the cabinet. One lamp indicates that the administrator has given a remote user the ability to dial into the system. And the second lamp indicates whether or not a remote access is in progress. If this lamp is flashing, the remote user has been given the same control as the system administrator.

DISTRIBUTED PROCESSING WITH PRIME SYSTEMS

The Prime 750, 650, and 550 support distributed processing with an array of software and hardware communications products. PRIMENET™, Prime's networking software, lets Prime computers communicate among themselves, with terminals, and with other manufacturers' systems over low-cost packet switching networks. Users can interface Prime computers to a variety of terminals and communications lines with multiple protocols and remote job entry options: IBM BISYNC for HASP and 2780; High-level Data Link Control (HDLC) protocol for X.25 packet switching networks; Control Data 200UT; Univac 1004; and ICL 7020. Prime's Distributed Processing Terminal Executive (DPTX) software conforms to protocols used by IBM 3271/3277 Display Systems. And Prime offers hardware controllers for both synchronous and asynchronous communications.

PRIME 750

PRIME

PRIME Computer, Inc., 40 Walnut Street, Wellesley Hills, MA 02181

Sales and service offices in major U.S. cities with subsidiaries in Denmark, Federal Republic of Germany, France, Norway, Sweden, and the United Kingdom; distributors in Australia, Austria, Canada, Ecuador, Greece, Italy, Japan, Korea, The Netherlands, New Zealand, Saudi Arabia, Singapore, and Switzerland.